4-13-2017 12:00 AM

Patch-based Denoising Algorithms for Single and Multi-view Images

Monagi H. Alkinani, *The University of Western Ontario*

Supervisor: Prof. Mahmoud R. El-Sakka, *The University of Western Ontario*

A thesis submitted in partial fulfillment of the requirements for the Doctor of Philosophy degree in Computer Science

Abstract

In general, all single and multi-view digital images are captured using sensors, where they are often contaminated with noise, which is an undesired random signal. Such noise can also be produced during transmission or by lossy image compression. Reducing the noise and enhancing those images is among the fundamental digital image processing tasks. Improving the performance of image denoising methods, would greatly contribute to single or multi-view image processing techniques, e.g. segmentation, computing disparity maps, etc. Patch-based denoising methods have recently emerged as the state-of-the-art denoising approaches for various additive noise levels. This thesis proposes two patch-based denoising methods for single and multi-view images, respectively.

A modification to the block matching 3D algorithm is proposed for single image denoising. An adaptive collaborative thresholding filter is proposed which consists of a classification map and a set of various thresholding levels and operators. These are exploited when the collaborative hard-thresholding step is applied. Moreover, the collaborative Wiener filtering is improved by assigning greater weight when dealing with similar patches.

For the denoising of multi-view images, this thesis proposes algorithms that takes a pair of noisy images captured from two different directions at the same time (stereoscopic images). The structural, maximum difference or the singular value decomposition-based similarity metrics is utilized for identifying locations of similar search windows in the input images. The non-local means algorithm is adapted for filtering these noisy multi-view images.

The performance of both methods have been evaluated both quantitatively and qualitatively through a number of experiments using the peak signal-to-noise ratio and the mean structural similarity measure. Experimental results show that the proposed algorithm for single image denoising outperforms the original block matching 3D algorithm at various noise levels. Moreover, the proposed algorithm for multi-view image denoising can effectively reduce noise and assist to estimate more accurate disparity maps at various noise levels.

Keywords: Patch-based image denoising, stereoscopic images, bilateral filter, non-local means filtering, probabilistic patch-based, dictionary learning, K-SVD, Gaussian patch-PCA, BM3D

Epigraph

"*If I can't picture it, I can't understand it.*"

— Albert Einstein

Dedication

To the woman who enfolds me—

MY PRECIOUS MOTHER, MALEHA,

who's been the greatest, and most lasting teacher throughout my life,

and to

THE MEMORY OF MY FATHER, HASAN,

whose faithful spirit sustains me still

Acknowledgements

First and foremost, I would like to thank Allah for all his blessings to me. You give me each day the power to undertake this work and complete it satisfactorily. I could never have done this without your blessings, Almighty.

I would like to express my sincere appreciation to Dr. Mahmoud R. El-Sakka for his heartfelt guidance, assistance, and supervision to pursue my research. I am immensely grateful for studying under each of your guidance.

I would like to thank my family for all endless encouragement while working on my thesis. Without their love, tremendous support and patience, it would not have been possible to accomplish my goals.

I acknowledge Jeddah University and the Cultural Bureau of Saudi Arabia in Canada for their advising and their financial support.

Table of Contents

List of Tables

List of Figures

List of Algorithms

List of Appendices

List of Abbreviations

ADF	Anisotropic Diffusion Filter
AWGN	Additive White Gaussian Noise
BF	Bilateral Filtering
BM	Block Matching
BM3D	Block Matching 3D
BM3D-Hou	BM3D of Hou
BM3D-SAPCA	Shape-Adaptive Principal Component Analysis
BM3D-SH2D	BM3D Sharpening 2D
BM3D-SSBS	BM3D with Smooth Sigmoid Shrinkage function
BM3D-UPSA	BM3D Iterative Up-sampling Algorithm
CPA	Chebyshev Polynomial Approximation
CWT	Continuous Wavelet Transform
DCT	Discrete Cosine Transform
DL	Dictionary Learning
DWT	Discretized Wavelet Transform
EPLL	Extend Patch Log Likelihood
GLCM	Grey-Level Co-occurrence Matrix
HSI	Hyper-spectral Image
HT	Hard Thresholding
HVS	Human Visual System
It-PPB	Iterative Probabilistic Patch-Based filter
K-SVD	K-means Singular Value Decomposition
KoK	Keep or Kill
LMSE	Laplacian Mean Square Error
LPA-ICI	Local Polynomial Approximation with the Intersection of the Confidence Interval
MAE	Mean Average Error
MAP-MRF	Maximum A Posteriori-Markov Random Field

MD	Maximum Difference
MLE	Maximum Likelihood Estimator
MSE	Mean Square Error
MSSIM	Mean Structural SIMilarity
NAE	Normalized Absolute Error
NCC	Normalized Cross-Correlation
NL-Means	Non-local Means
NLF	Noise Level Function
Non-iPPB	Non-iterative Probabilistic Patch-Based filter
PB-PCA	Patch-based Principal Component Analysis
PCA	Principal Component Analysis
PGPCA	Patch-based Global PCA
PHPCA	Patch-based Hierarchical PCA
PLPCA	Patch-based Local PCA
POL-SAR	Polarimetric Synthetic Aperture Radar
PPB	Probabilistic Patch-Based filter
PPBWE	PPB Weights Estimator
PSNR	Peak Signal-to-Noise Ratio
RGB	Red, Green, and Blue
S-MD	Non-local Means with Maximum Difference similarity for Stereo image denoising
S-SSIM	Non-local Means with Structural similarity for Stereo image denoising
S-SVD	Non-local Means with SVD-based similarity for Stereo image denoising
SC	Structural Content
SNR	Signal Noise Ratio
SSBS	Smooth Sigmoid Shrinkage Function
SSIM	Structural SIMilarity
ST	Soft Thresholding
Stereo-MSE	Stereo denoising with Mean Square Error
SVD	Singular Value Decomposition
SVD-based	Singular Value Decomposition-based similarity assessment
TV	Total Variation
USC-SIPI	University of Southern California-Signal and Image Processing Institute
WMLE	Weighted Maximum Likelihood Estimation
XNLM	X-ray Non-local Means

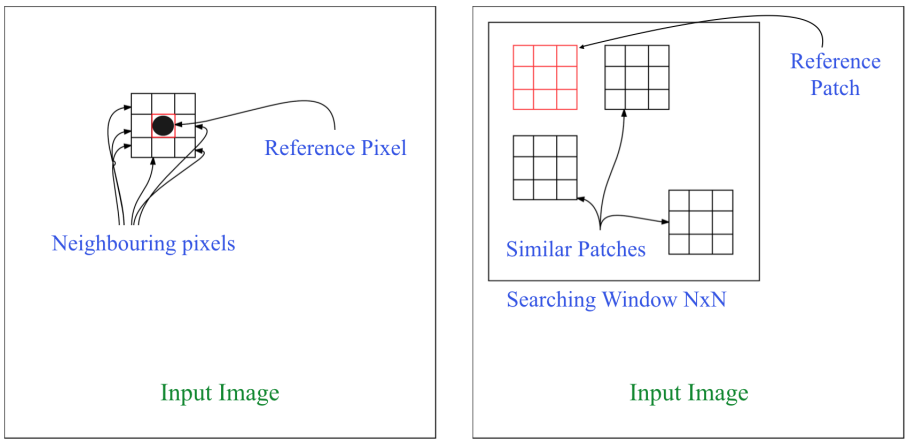
Chapter 1

Introduction

Dealing with random noisy pixels in an image is a major challenge for many image processing applications due to the difficulty of distinguishing noise from true image information. Noise affects all kinds of images whether they are single or multi-view. It also affects various types of image areas, such as textured, flat or edge areas. Image denoising techniques, which can estimate the true image pixels, are broadly grouped into two main approaches: pixel-based filtering and patch-based filtering. A pixel-based image filtering scheme is mainly a proximity operation used for manipulating one pixel at a time (pixel-wise). It is based on examining spatial neighbouring pixels located within a kernel. On the other hand, in patch-based image filtering, the noisy image is divided into patches, or "blocks", which are then manipulated separately in order to provide an estimate of the true pixel values (patch-wise). It is based on similar patches located within a search window. Unlike pixel-based image filtering, patch-based and algorithms successfully utilizes the redundancy and the similarity among the various parts of the input image to reconstruct a more distinct image. The mechanisms of using these two methods are illustrated in Figure 1.1.

1.1 Problem Background

Single and multi-view images are captured using sensors during the data acquisition phase. A single view image consists of one image; a multi-view image consists of two or more images, which are generated simultaneously from different camera sources at different locations, and are focused on one scene. These images are often contaminated with undesirable random noise during image acquisition, transmission, or by poor image compression.



(a) (b)

Figure 1.1: Image denoising categories: (a) filtering based on neighbouring pixels located within a kernel in pixel-based denoising schemes, and (b) filtering based on patches located within a searching window in patch-based denoising schemes.

Single images are used in diverse image processing applications, such as in object recognition, segmentation, and object tracking. Noisy single images exert significant influence on the reliability of these applications. For insistence, when detecting non-smoothed objects in noisy images, some algorithms may mistakenly detect disconnected objects due to the presence of noisy pixels. Figure 1.2 shows the results of an attempt to detect coins using snakes active contours [59]. The figure reveals that when noise contaminates the targeted objects "coins" they make it more difficult to produce reliable segmentations. Noise misleads the contours to not fit exactly onto object boundaries, see Figure 1.2 (d).

Multi-view images have even more sophisticated imaging applications. These applications include 3D stereoscopic film, and the extraction of depth information. The quality of these applications depends mainly on the image clarity. In support of this, the disparity maps[1] of a stereoscopic image both with and without noise were obtained using the graph-cut algorithm of Boykov *et al.* [16] in Figure 1.3. When the stereoscopic image is corrupted by noise, the extracted depth information is not clear, see Figure 1.3 (d).

The presence of noise in a single and/or a multi-view image has a significantly negative impact on the results of various image processing applications. Hence, finding effective ways of reducing noise in digital images is a very worthwhile area of research. Some noise reducing methods utilize neighbouring pixels in order to estimate a centre pixel [104, 95]; however, such methods fail to

[1]The human brain process binocular disparity (the difference in image location of an object seen by the left and right eyes) for extracting depth information from the two-dimensional retinal images. In computer vision, disparity map represents the difference in coordinates of similar objects within two digital stereoscopic images.

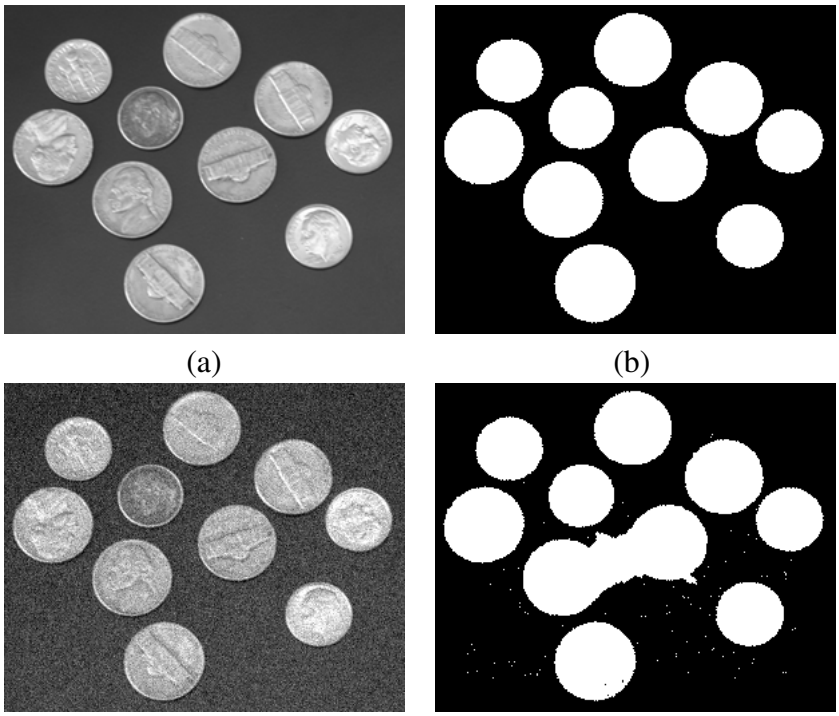


Figure 1.2: Our results of detecting coins using snakes active contours of Kass *et al.* [59]: (a) noise-free image, and (b) the result of detecting coins using the noise-free image, (c) noisy image with *additive white Gaussian noise* ($\sigma = 20$), and (d) the result of detecting coins when noise is present.

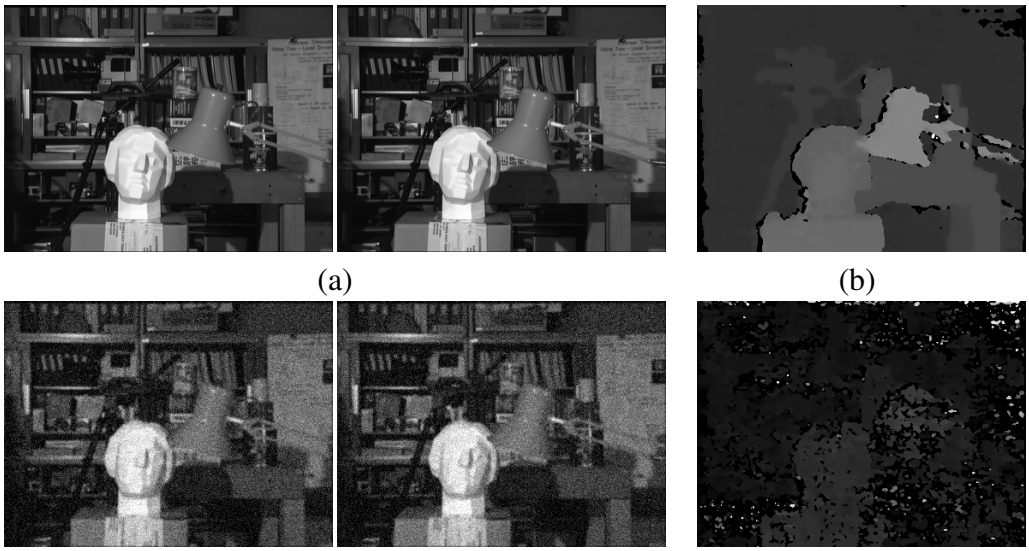


Figure 1.3: Our results of computing the disparity maps of stereoscopic images using the graph-cut algorithm of Boykov *et al.* [16]: (a) noise-free image stereoscopic image, (b) disparity maps of the noise-free stereoscopic image, (c) noisy image with *additive white Gaussian noise* ($\sigma = 20$), and (d) the result of computing disparity maps when noise is present.

reduce high levels of noise and retain image details due to high intensity variations. Several recent research studies employ similar patches to restore a noise-free patch [17, 32, 61, 24]. Such findings benefit from the redundancy, which appears between patches, but they fail to properly preserve fine details and smooth flat regions, and are very expensive in terms of time and memory when they are utilized for multi-view images. Thus, there is a need to develop a new patch-based image denoising method, which has the ability to preserve fine details and smooth properly flat regions, for single and multi-view images.

1.2 Problem Statement

The *Block matching 3D* (BM3D) filtering algorithm of Dabov *et al.* [24] is a patch-based denoising method, which depends mainly on the transformation coefficient thresholding procedure. When transferring a signal into the frequency domain, noise is characterized by low amplitude levels. The BM3D filtering algorithm uses a fixed thresholding operator to force all of the coefficients below the threshold to be zeros. This algorithm is considered to be a state-of-the-art denoising method. Yet this method has several disadvantages, such as that of the difficulties related to the choosing an optimal thresholding operator. When Marc Lebrun [64] described the influence of all of the twenty parameters of the BM3D, he concluded that the thresholding operator is one of the most important parameters of the BM3D because it controls the amount of the coefficient's thresholding level of the 3D patches group. Although previous researchers have shown ways of optimally adjusting the BM3D parameters, they have not shown them bing used as an adaptive thresholding operator with a textures classification map which considers geometric and luminance similarity distances. This study will address this classification map, in order to find an effective way of enhancing single view images.

In the case of multi-view image denoising, various image similarity measures will be addressed; these include *maximum difference*, *structural similarity* [97], and *singular value decomposition* similarity [79] measures, in order to identify similar search windows among the multi-view images. The *non-local means* (NL-Means) algorithm of Buades *et al.* [17] will then be adapted for multi-view image denoising.

The research objective is therefore to investigate whether improving the means of optimizing the thresholding operator could improve the overall ability of the BM3D method while preserving detail and suppressing noise in single view images. In addition, weight assignment for the Wiener filtering of BM3D will be addressed, and the effect of the transforms will be investigated. Fi-

nally, the effectiveness of adapting a NL-Means filtering algorithm using various image similarity measures for denoising multi-view images will be studied.

1.3 Research Purpose

Evaluating the geometric and luminance similarity distances between neighbouring pixels assist the pixel-based *bilateral filter* of Tomasi *et al.* [95] to optimally adjust the required weights before averaging pixels. By considering both similarity distances, this pixel-based filter preserves the desired detail and suppresses more noise. Hence, adapting this idea to the patch-based BM3D filter when thresholding the 3D group coefficients, we can preserve more detail and can suppress more noise in a noisy single view image if the differences between flat and textured areas were considered.

Increasing the number of similar patches before thresholding or averaging improves the effectiveness of the overall denoising procedure of the patch-based image denoising methods. Thus, instead of denoising each view separately in a multi-view image, this study extends the search window to use more reliable patch similarity measures in order to cover the entire spectrum of multi-view images. The effectiveness of several image similarity measures is examined to identify the location of similar windows.

1.4 Research Objectives

The objective of this research is to develop effective approaches for denoising single and multi-view digital images that fulfill the following requirements:

1. Exceed the performance of existing single and multi-view image denoising methods,
   (a) quantitatively via:
      i. The *peak signal-to-noise ratio* (PSNR).
      ii. The *mean structural similarity* (MSSIM) of Wang *et al.* [97].
   (b) qualitatively via:
      i. The preservation of textures and fine details.
      ii. A sharpening of the edges.

2. Extract greater depth of information over what is possible with the existing multi-view image denoising methods (via disparity maps).

3. Produce a solution with simple computational complexity when denoising multi-view images (via bounded searching areas).

By meeting these objectives, effective enhanced single or multi-view images would be achieved.

1.5 Research Methodology

In this section, the datasets and the techniques used to check the validity and reliability of this work are described. Two image databases were used to evaluate the performance of the denoising methods: a single image database, and a multi-view image database.

The single image dataset contains images obtained from the image database of the *University of Southern California-Signal and Image Processing Institute* (USC-SIPI) [1]. The USC-SIPI's database has more than 237 images, but this research uses only ten images that were used in the original BM3D research. This research ensured that the chosen images contain fine details, sharp edges, grey gradations and pattern repetitions. The fine details of the `Lena`, `Barbara`, `Man`, and `Boats` images were helpful in demonstrating how the various methods can preserve the image clarity; whereas the sharp edges of the `House` and `Cameraman` were helpful in demonstrating how the various methods can preserve the edges. The grey gradations of `Hill` image provided insight into the amount of smoothing that has been applied to the images, and the `Fingerprint` image assisted in revealing how pattern repetitions were preserved by the various methods.The size of the single view images varies, e.g., 256×256 pixels, or 512×512 pixels. And they are all grey-scale images. That is, they have 8 bits per pixel.

A collection of multi-view images from the stereoscopic database of *Middlebury Computer Vision* [3] was used to evaluate the multi-view image denoising methods. Middlebury stereoscopic database has 71 images, but four multi-view images were used in this work. The images have been carefully chosen to assist in distinguishing between the qualities of output from the various performance methods. The house in `Teddy`'s image assist in demonstrating how methods preserve sharp edges. Head and lamp in the `Tuskuba`'s image help in evaluating how methods smooth properly homogeneous regions. The grey gradations in the `Cones` image provided insight into the amount of smoothing. The texts in the `Newspaper` images assist in studying the ability to preserving fine

details. The size of the images varies, e.g., 450×375 pixels, 434×383 pixels, or 384×288 pixels. And they are all grey-scale images represented as 8 bits per pixel.

The image datasets were corrupted using *additive white Gaussian noise* (AWGN) with a noise variance that ranged from 10 to 100 σ. The denoising methods then were employed to estimate the free of noise (a true image). This thesis exploits full reference similarity measures. The reference image is assumed to be in existence and free of noise, as compared with the denoised image.

1.6 Thesis Structure

This thesis contains six chapters and three appendices. In Chapter 1, an introduction to image denoising is introduced. In Chapter 2, the background and the literature review on the subject of image filtering are explained. In Chapter 3, the empirical results and discussion are produced. In Chapter 4, the proposed patch-based denoising method for single image denoising is presented and discussed. In Chapter 5, patch-based denoising methods for multi-view image denoising are presented and clarified. In Chapter 6, an empirical evaluation and the results of the denoising methods are provided. In Chapter 7, the conclusions and a perspective on the need for future work on the subject of developing new patch-based image denoising methods are addressed. Finally, Appendix A, B, C, and D provide greater results on the denoising methods comparison.

Chapter 2

Images Denoising: Background and Literature Review

In this chapter, a general introduction to digital image denoising is presented. This chapter is divided into five sections. In Section 2.1, the main noise models are presented. In Section 2.2, a background on single image filtering is introduced. In Section 2.3, a background on multi-view image filtering is provided. Finally, the image similarity measurements are described in Section 2.5.

2.1 Types of Noise in Digital Images

Noise in digital images could be additive, multiplicative, or a mixture of such errors occurring in the original signal. Additive noise is an interference added to the signal image pixel. Such noise in a digital image is generally modelled as

$$\upsilon(x) = u(x) + n(x)_d, x \in \Omega, \tag{2.1}$$

where $\upsilon(x)$ is the noisy component of the image, $u(x)$ is the true image, $n(x)_d$ is the random additive noise, and Ω denotes the set of all pixels in the image. In particular, if $n(x)_d$ is a Gaussian random process, the noise is identified as Gaussian noise.

Multiplicative noise, which is also called speckle noise, refers to the interference that is introduced

into the signal. Multiplicative noise in a digital image is modelled as:

$$\upsilon(x) = u(x) \times n(x)_m, x \in \Omega \tag{2.2}$$

where $\upsilon(x)$ is the noisy component of the image, $u(x)$ is the true image, $n(x)_m$ is the random multiplicative noise, and Ω denotes the set of all pixels in the image.

The combination of the additive and multiplicative noises is called mixed noise. Mixed noise in a digital image is modelled as:

$$\upsilon(x) = n(x)_m \times u(x) + n(x)_d, x \in \Omega \tag{2.3}$$

where $\upsilon(x)$ is the noisy component of the image, $u(x)$ is the true image, $n(x)_d$ is the random additive noise, $n(x)_m$ is the random multiplicative noise, and Ω denotes the set of all pixels in the image.

Image denoising methods can be categorized into two main groups: single image denoising methods and multi-view image denoising methods. While one image is processed in a single image denoising process, multiple images are processed in a multi-view image denoising process.

2.2 Single Image Denoising

2.2.1 Pixel-based Image Filtering

In this subsection, a short background on pixel-based image filtering, which is an operation used for manipulating one pixel at a time, is provided. Through the filtering process, a revised pixel value is calculated using coordinates that are equal to the coordinates of the centre of a kernel. The new pixel value is the result of the filtering process. When the centre of the filter visits each pixel of the input image, a filtered image is generated. Linear filters then replace each image pixel with a linear combination of its neighbours. Otherwise, the filter is called a nonlinear filter.

By assuming $H(x, y)$ is a kernel of size $m \times n$ with the coordinates (x, y), where m and n are equal to $2a + 1$ and $2b + 1$, respectively, and a as well as b are positive integers, $I(x, y)$ is an input image with size $M \times N$, and $\hat{u}(x, y)$ is a filtered image with size $M \times N$, the filtering process for the image is given by the expression:

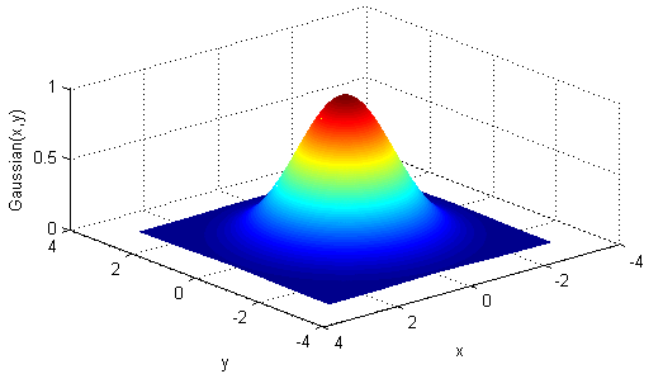


Figure 2.1: 2D Gaussian distribution with mean (0,0) and σ=1

$$\hat{u}\,(x,y) = \sum_{i=-a}^{a}\sum_{j=-b}^{b} I(x+i, y+j)\cdot H(i,j) \tag{2.4}$$

2.2.1.1 Low-pass Filtering

Since noise in digital images is considered to be a high frequency component, applying low-pass filters can eliminate noise in such images. The strength of the filters is related to the kernel sizes and weights. There are several low-pass filters, e.g., mean, Gaussian and Yaroslavsky filters.

- Mean Filter: A mean filter replaces each pixel value with the average value of itself and its neighbours. By considering Equation 2.4, the kernel of the mean filtering for an image is provided in the expression:

$$H(i,j)_{\text{Mean}} = \frac{1}{m \times n} \tag{2.5}$$

- Gaussian Filter: A Gaussian filter is similar to the mean filter, but it uses a kernel that represents the shape of a Gaussian hump. A Gaussian filter could be described as a weighted average filter of the nearest pixels' neighbours. Thus, it produces a smoother image than that produced by a similarly sized mean filter. The definition of the weight is the main issue to be addressed in averaging an additive noise for denoising. The weight in a Gaussian filter is defined locally based on the neighbouring pixel's location. The weights increase when neighbours are closer to the central pixel's location. The 2D Gaussian distribution with mean (0,0) and $\sigma = 1$ is provided in Figure 2.1 below. The Gaussian low-pass filter

$H(i,j)_{\text{Gaussian}}$ has the form of Equation 2.6; the weight depends on the geometric distance between pixels,

$$H(i,j)_{\text{Gaussian}} = \frac{1}{\sqrt{2\pi}\sigma_d} e^{-\frac{|i-j|^2}{2\sigma_d^2}} \tag{2.6}$$

where i, j are the coordinates of a pixel ($i, j \in \Omega$), σ_d is the standard deviation of the Gaussian hump that determines the degree of similarity between spatially close pixels. The ideal Gaussian filter has the property of having values less than one standard deviation from the mean represents about 68.3% of the data under the Gaussian hump; two standard deviations represent about 95.5% of the data, and three standard deviations represent about 99.7% of the data. The standard deviation of the Gaussian filter controls the degree of smoothing. The standard deviation value is associated with the kernel size. A high standard deviation value with a small kernel size makes a Gaussian filter behave more like a mean filter, and hence it produces more smoothing of the edges. Figure 2.2 shows deploying various standard deviations (σ) for denoising `Lena` image contaminated with AWGN. The edges are blurred when high values are used for the standard deviation with the same kernel size. The plot in Figure 2.3 illustrates the effect of the Gaussian filters that are used in Figures 2.2(d), (e) and (f) on preserving edges. The grey values of the columns, which range from 105 to 150 in row 150 of `Lena` image, are plotted in order to provide a visualization of the amount of edge blurring. From the plot, the edges are blurred as the standard deviation value increases.

- Yaroslavsky Filter: With a Yaroslavsky filter [104], the pixels are restored using the weighted average value of the pixels with similar grey level values within the same spatial neighbourhood. This type of filter is an image-dependent filter; hence, the weights cannot be defined using an image independent kernel. If we have a spatial neighbourhood in a kernel of size $n \times m$, the Yaroslavsky filter is defined as the formula shown in Equation 2.7,

$$H(I(x,y),I(i,j))_{\text{Yaroslavsky}} = \frac{1}{\sqrt{2\pi}\sigma_r} e^{-\frac{|I(x,y)-I(i,j)|^2}{2\sigma_r^2}} \tag{2.7}$$

where $I(x,y)$ and $I(i,j)$ are the grey levels at two positions, and σ_r is the standard deviation of the Gaussian hump that determines the degree of similarity between the grey levels.

2.2.1.2 Bilateral Filtering

The basic idea underlying *bilateral filter* (BF) is to prevent the averaging across edges, while performing averaging within homogeneous regions [95]. While the weighted average in a Gaussian

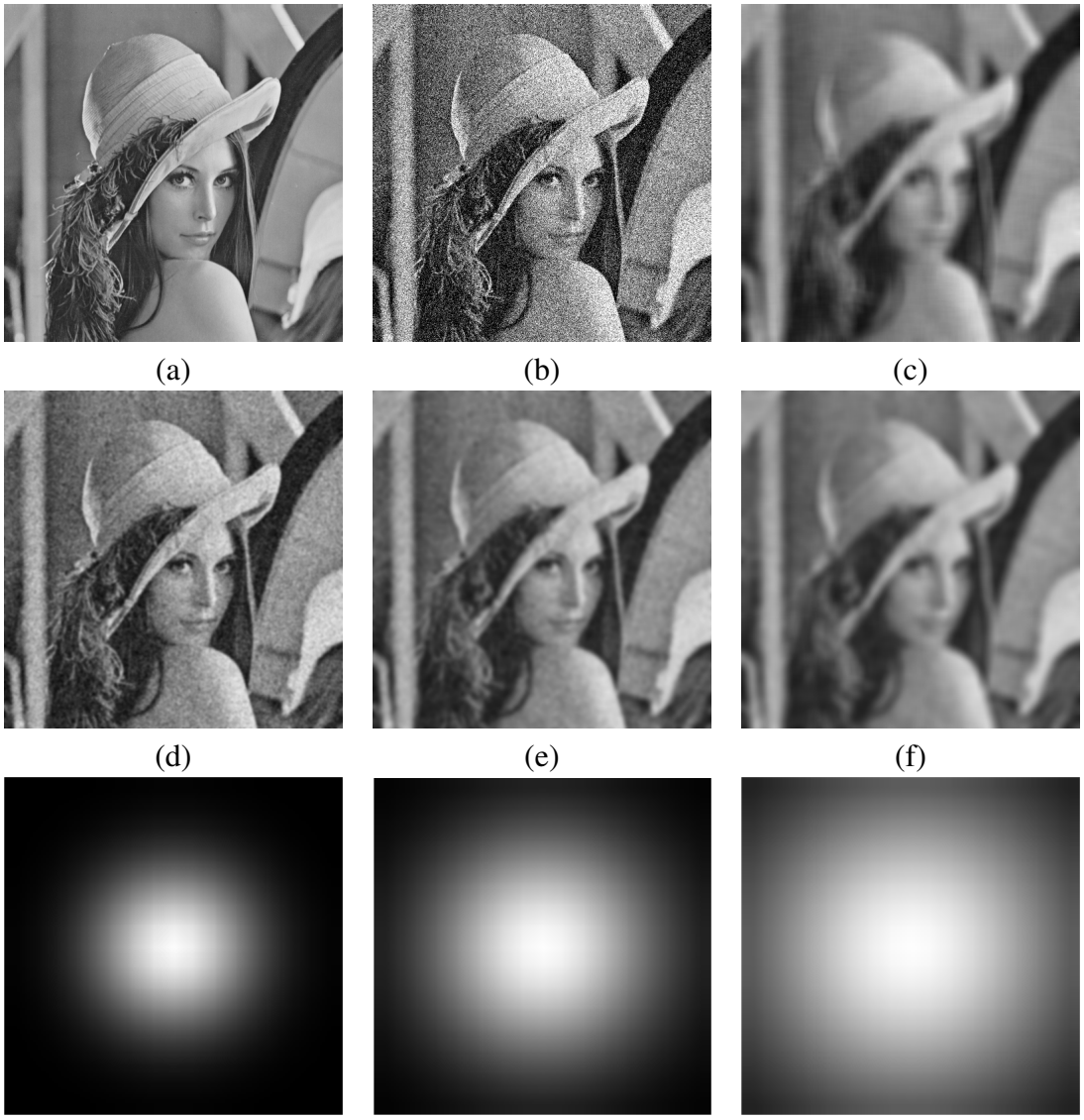
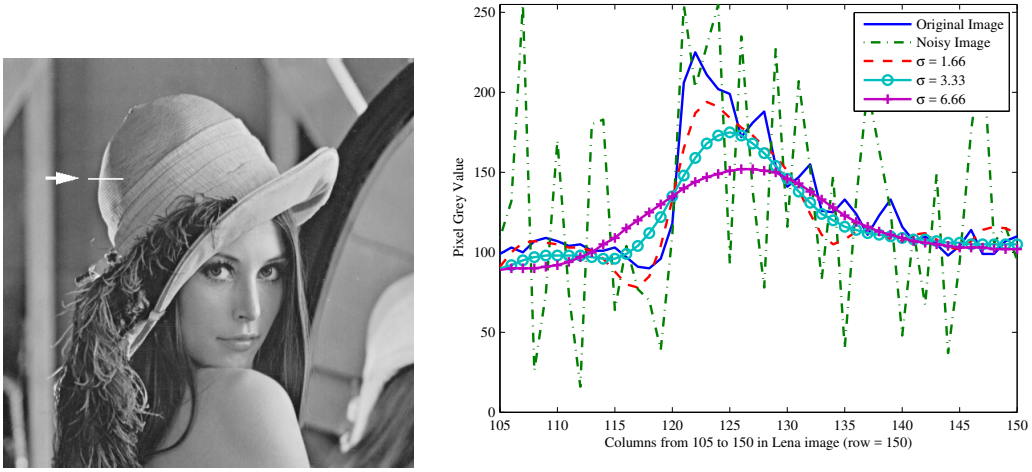


Figure 2.2: A Gaussian filter has different sigma values: (a) an original *Lena* image, (b) an AWGN noisy image with $\sigma = 20$, (c) a denoised image using a 21×21 mean filter, (d) a denoised image using a 21×21 Gaussian filter $\sigma = 1.66$, (e) a denoised image using a 21×21 Gaussian filter $\sigma = 3.33$, and (f) a denoised image using a 21×21 Gaussian filter $\sigma = 6.66$. The third row shows the 2D Gaussian kernels used in the second row filters.



(a) (b)

Figure 2.3: The effect of the standard deviation values of the Gaussian filters used in Figure 2.2: (a) position of the columns from 105 to 150 in row 150, and (b) plot of the grey values of the columns.

filter is based on the geometric distance between pixels, the weighted average in a Yaroslavsky filter is based on the luminance distance between pixels. BF utilizes these two distances, and assigns higher weight to similar pixels. Hence, the weight of two pixels could be similar if they were to be spatially located within the same distance from the centre pixel and they would have perceptually similar values.

Since BF measures the geometric distance similarity using Equation 2.6 and the luminance similarity using Equation 2.7 for computing weights, the BF is shown as:

$$\hat{u}(x,y) = \sum_{i=-a}^{a}\sum_{j=-b}^{b} I(x+i, y+j) \cdot H(i,j)_{\text{Gaussian}} \cdot H(I(x,y), I(i,j))_{\text{Yaroslavsky}} \tag{2.8}$$

The weight $H(I(x,y), I(i,j))_{\text{Yaroslavsky}}$ prevents averaging across edges. The pixels belonging to the same region are weighted averages, if the grey level difference between the two pixels is small. Therefore, the BF does not blur the edges. The BF has the ability to average across features that are located within $2\sigma_d$ from the centred pixel while preserving the fine details. Figure 2.4 shows the robustness of a BF in preserving the fine details.

The run-time of BF limits its usefulness in real-time. The BF is ineffective in smoothing regions with high gradients; Figure 2.5 (a) provides an example of such signals. BF fails when there are not enough pixels to average. For example, presume that the grey level of a local neighbourhood consists of some narrow ridges separated by narrow valleys as shown in Figure 2.5 (b). As any

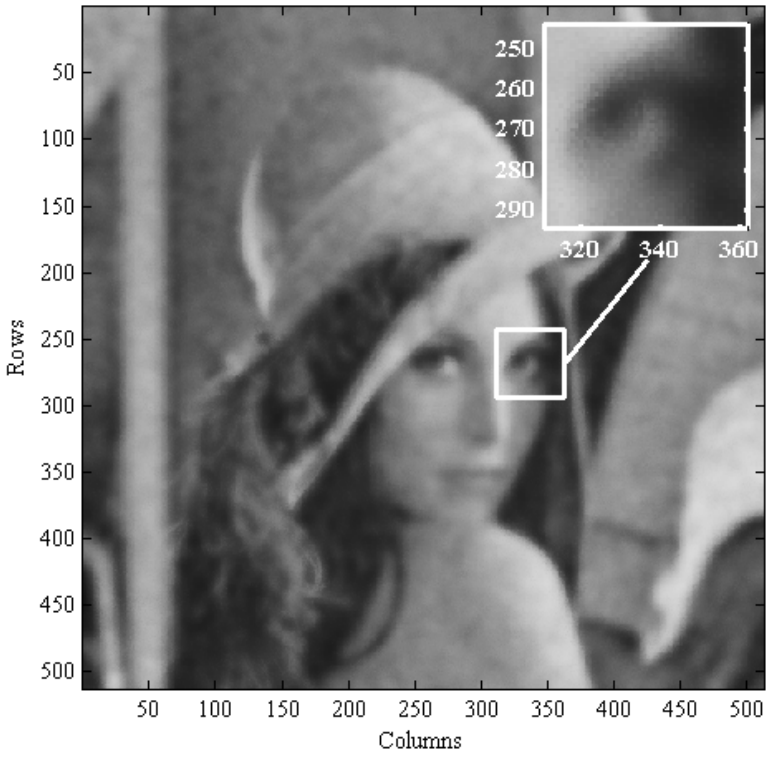
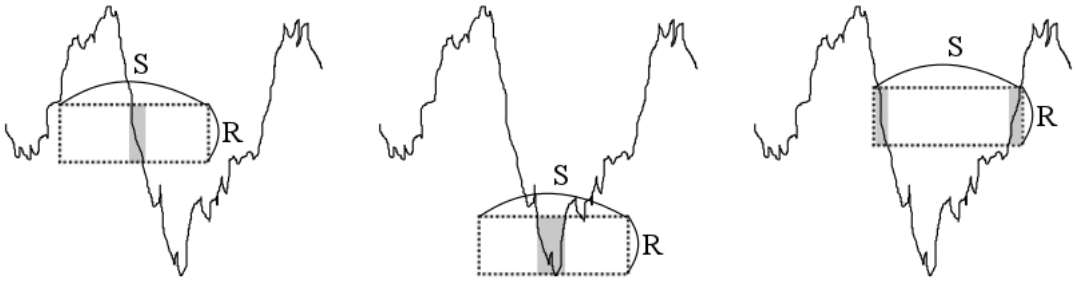


Figure 2.4: Robustness of the BF. The BF preserves the edges and the fine details. The close-up image shows the preserving of fine detail.



(a) (b) (c)

Figure 2.5: BF in one-dimensional signals of different local neighbourhood grey levels: (a) area with high gradient, (b) area with ridges and valleys, and (c) disjoint region. Where S is the geometric distance similarity and R is the luminance similarity range.

non-adaptive spatial filter, Figure 2.5 (c) shows that the BF fails to separate the disjointed regions. The BF cannot be effectively adapted to filtering out these local features. All of the filters addressed so far in this subsection (Subsection section 2.2.1) are non-iterative filters, which produces a result in a single pass. All of the remaining filters are iterative filters.

2.2.1.3 Variational Denoising Methods

The goal of variational denoising methods is to minimize the high level of energy in noisy images. In order to minimize the high level of energy, regularization in variational denoising permits the selection of the most reasonable solution from a group of several solutions. By assuming $\upsilon(x, y)$ is a noisy image and $E(\upsilon)$ is the energy that describes the quality of the image, the energy minimization process to estimate the original image $\hat{u}(x, y)$ is defined as:

$$\hat{u}(x, y) = \arg\min E(\upsilon) \tag{2.9}$$

Finding the local minimum of the energy in Equation 2.9 can be achieved by using a gradient descent optimizer. The gradient descent optimizer is an optimizer that takes a small step toward the negative direction of the energy gradient $(-\nabla E)$. The gradient descent is defined as:

$$\hat{u}(x, y)^{t+1} = \hat{u}(x, y)^{t} + \Delta t(-\nabla E) \tag{2.10}$$

where $\hat{u}(x, y)^{t}$ is the current pixel value, $\hat{u}(x, y)^{t+1}$ is the new pixel value and Δt is the step size.

This optimizer is utilized below in minimizing the heat diffusion and the total variation problems.

- Isotropic Heat Diffusion: A sudden increase in a pixel value indicates a noisy pixel. Thus, the gradient $|\nabla \upsilon(x, y)|$ is expected to be high near a noisy pixel. In order to smooth the gradient values, the isotopic heat diffusion equation could be used as the energy to be minimized. The heat equation is a parabolic partial differential equation, used to describe the behaviour of the heat distribution in a region over time. If $q(x, y)$ is a function of two spatial variables (x, y) and t is the time variable, the heat equation formula is formalized as:

$$\frac{\partial q}{\partial t} - \alpha \left(\frac{\partial^2 q}{\partial x^2} + \frac{\partial^2 q}{\partial y^2} \right) = 0 \tag{2.11}$$

  where α is a positive constant that controls the amount of the applied diffusion, and $\left(\frac{\partial^2 q}{\partial x^2} + \frac{\partial^2 q}{\partial y^2}\right)$ is the laplacian operator $(\triangle q)$. The heat equation diffuses equally into all directions. Thus, it is called an isotopic diffusion equation. For the filtering of a noisy image $\upsilon(x, y)$, Equation 2.11 is used as the energy gradient function ∇E optimized by the gradient descent shown in Equation 2.10. The heat diffusion energy for the image is discretized as:

$$\nabla E = \triangle \upsilon = \left(\nabla^{xx} \upsilon(x, y)^t + \nabla^{yy} \upsilon(x, y)^t \right) \tag{2.12}$$

  where $\nabla^{xx} \upsilon(x, y)^t$ and $\nabla^{yy} \upsilon(x, y)^t$ are:

$$\begin{aligned} \nabla^{xx} \upsilon(x, y)^t &\approx \upsilon(x+1, y)^t - 2\upsilon(x, y)^t + \upsilon(x-1, y)^t \\ \nabla^{yy} \upsilon(x, y)^t &\approx \upsilon(x, y+1)^t - 2\upsilon(x, y)^t + \upsilon(x, y-1)^t \end{aligned} \tag{2.13}$$

  The heat diffusion imparts more smoothing to the image as the step size or the number of iterations increases. The minimization process may fail to converge to a local minima when using large step sizes. It is recommended that the step size be $(0 \leq \triangle t \leq 1/4)$ [83]. Figure 2.6 shows the result of using various steps sizes along with the isotopic heat diffusion for image filtering. In order to prevent smoothing across edges, a condition is needed to force the heat equation to be $\frac{\partial \upsilon}{\partial t} = 0$ near boundaries. A *total variation* (TV) filter uses a data fidelity term to overcome this problem.

- Total Variation: Rudin *et al.* [90] introduced the TV regularization for image denoising. The TV method is an edge-preserving denoising method. The TV method consists of two terms: a regularization term and a data fidelity term. The regularization term assists in smoothing flat regions, while the data fidelity term is used as a condition to prevent the smoothing of the cross-image features. By assuming that $\upsilon(x, y)$ is a noisy image, the original image $u(x, y)$ could be restored as the solution to the regularization problem

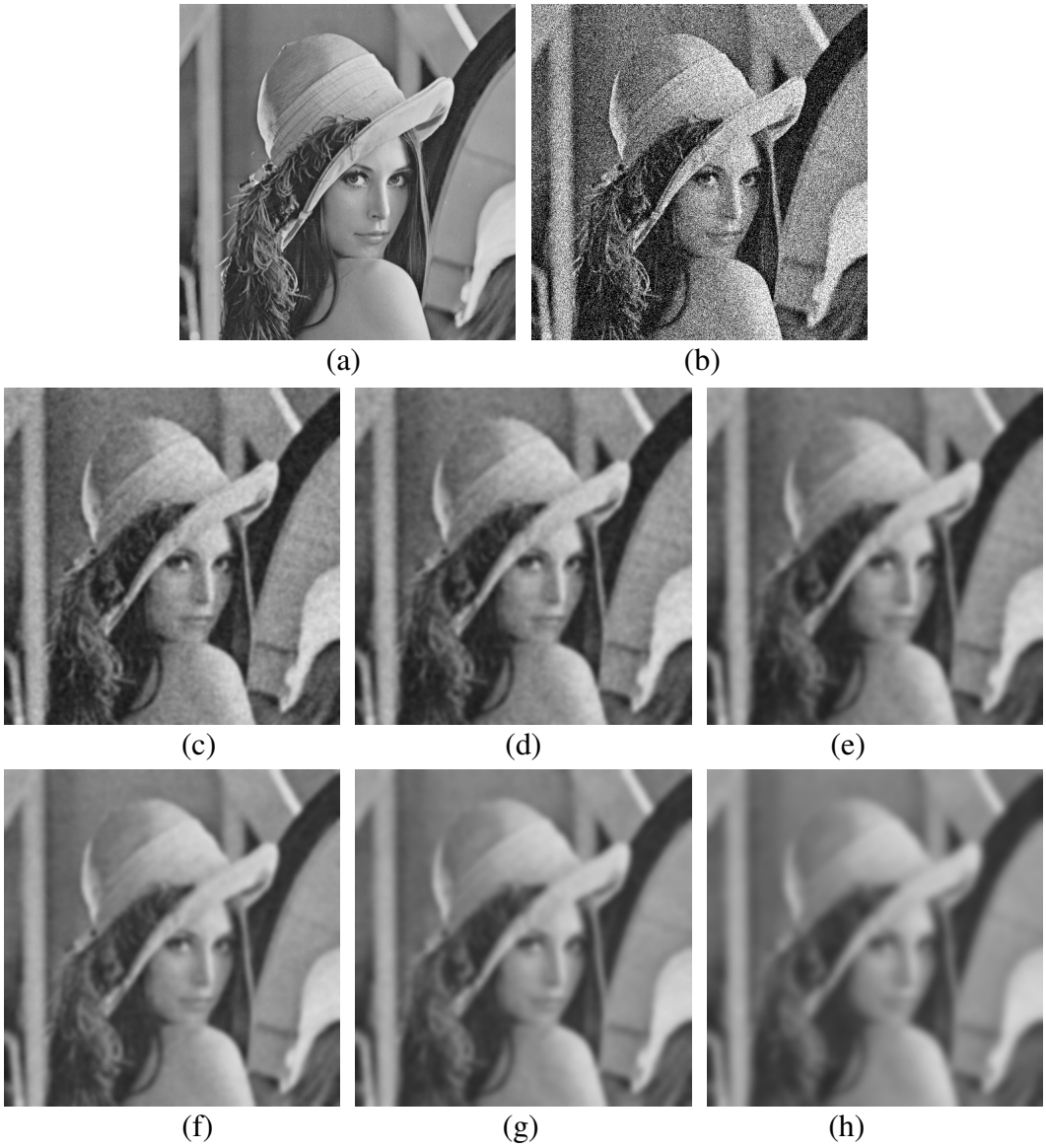


Figure 2.6: Isotropic heat diffusion filtering: (a) an original *Lena* image, (b) an AWGN noisy image with σ = 20, (c) a denoised image using heat diffusion (*iteration* = 30, $\triangle t$ = 0.05), (d) a denoised image using heat diffusion (*iteration* = 50, $\triangle t$ = 0.05), (e) a denoised image using heat diffusion (*iteration* = 100, $\triangle t$ = 0.05), (f) a denoised image using heat diffusion (*iteration* = 30, $\triangle t$ = 0.2), (g) a denoised image using heat diffusion (*iteration* = 50, $\triangle t$ = 0.2), and (h) a denoised image using heat diffusion (*iteration* = 100, $\triangle t$ = 0.2).

$$\arg\min_{\upsilon} \underbrace{TV(\upsilon)}_{\text{Regularization Term}} + \underbrace{\lambda \cdot |u-\upsilon|^2}_{\text{Data Fidelity Term}} \tag{2.14}$$

where $TV(\upsilon)$ indicates the TV and $\lambda > 0$ is a parameter that balances the importance of the two terms in order to control the degree of filtering. The numerical approximation of the energy shown in Equation 2.14 is:

$$\begin{aligned} \nabla E &= \frac{1}{h}\left[\left(\frac{\nabla_+^x \upsilon(x,y)^t}{\sqrt{\left(\nabla_+^x \upsilon(x,y)^t\right)^2 + \left(m\left(\nabla_+^y \upsilon(x,y)^t, \nabla_-^y \upsilon(x,y)^t\right)\right)^2}}\right)\right. \\ &\left. + \left(\frac{\nabla_+^y \upsilon(x,y)^t}{\sqrt{\left(\nabla_+^y \upsilon(x,y)^t\right)^2 + \left(m\left(\nabla_+^x \upsilon(x,y)^t, \nabla_-^x \upsilon(x,y)^t\right)\right)^2}}\right)\right] \\ &-\lambda^t\left(\upsilon(x,y)^t - \upsilon_0(x,y)\right) \end{aligned} \tag{2.15}$$

here, the neighbour differences ($\bigtriangledown$) are computed as

$$\begin{aligned} \nabla_+^x \upsilon(x,y)^t &\approx \upsilon(x+1,y)^t - \upsilon(x,y)^t \\ \nabla_-^x \upsilon(x,y)^t &\approx \upsilon(x-1,y)^t - \upsilon(x,y)^t \\ \nabla_+^y \upsilon(x,y)^t &\approx \upsilon(x,y+1)^t - \upsilon(x,y)^t \\ \nabla_+^y \upsilon(x,y)^t &\approx \upsilon(x,y-1)^t - \upsilon(x,y)^t \end{aligned} \tag{2.16}$$

$m(\cdot)$ is computed as

$$\begin{aligned} m(A,B) &= minmod(A,B) \\ &= \left(\frac{sgn\ A + sgn\ B}{2}\right) min\left(|A|, |B|\right) \end{aligned} \tag{2.17}$$

and λ^t is defined via

$$\begin{aligned}\lambda^t &= \frac{h}{2\sigma^2}\left[\sum_{(x,y)}\left(\sqrt{\left(\nabla_+^x \upsilon\,(x,y)^t\right)^2 + \left(\nabla_+^y \upsilon\,(x,y)^t\right)^2}\right)\right. \\ &\quad -\left(\frac{\nabla_+^x \upsilon\,(x,y)^0 + \nabla_+^x \upsilon\,(x,y)^t}{\sqrt{\left(\nabla_+^x \upsilon\,(x,y)^t\right)^2 + \left(\nabla_+^y \upsilon\,(x,y)^t\right)^2}}\right) \\ &\quad \left.-\left(\frac{\nabla_+^y \upsilon\,(x,y)^0 + \nabla_+^y k \cdot \upsilon\,(x,y)^t}{\sqrt{\left(\nabla_+^x \upsilon\,(x,y)^t\right)^2 + \left(\nabla_+^y \upsilon\,(x,y)^t\right)^2}}\right)\right]\end{aligned} \tag{2.18}$$

where σ is the noise estimated variance and h, is the number of neighbours, which is equal to 4 for the 4-nearest neighbours or to 8 for the 8-nearest neighbours.

For image filtering, the energy function shown in Equation 2.15 is optimized by the gradient descent shown in Equation 2.10. In order to analyze the energy convergence in the TV method, two types of convergence are used with the gradient descent optimization in this thesis. One of the convergences utilizes a fixed number of iterations, while the rest of the convergences automatically calculates the number of iterations using the the *mean square error* as a condition. The automatic convergence stops if the *mean square error* between two iterations is less than 0.01, which means there is no visual quality improvement could be achieved. The results of utilizing the two convergence types for the denoising of `Lena` image are shown in Figure 2.7. Figures 2.7 (c), (d), and (e) show that increasing the iteration numbers imparts more smoothing to the images. Figures 2.7 (f), (g), and (h) show that textures are blurred when the λ value decreases.

2.2.1.4 Anisotropic Diffusion Filtering

The *anisotropic diffusion filter* (ADF), introduced by Perona and Malik [83], iteratively blurs the non-edge regions while keeping the edges and the boundaries sharp. The ADF does not apply smoothing the same way in different directions; thus, it is called an-iso-tropic method. The anisotropic diffusion equation for image denoising is defined as

$$\begin{aligned}\hat{u}\,(x,y)^{t+1} &= div\left(c\,(x,y,t)\bigtriangledown \upsilon\,(x,y)\right) \\ &= c\,(x,y,t)\bigtriangleup \upsilon\,(x,y) + \bigtriangledown c\,(x,y,t)\bigtriangledown \upsilon\,(x,y)\end{aligned} \tag{2.19}$$

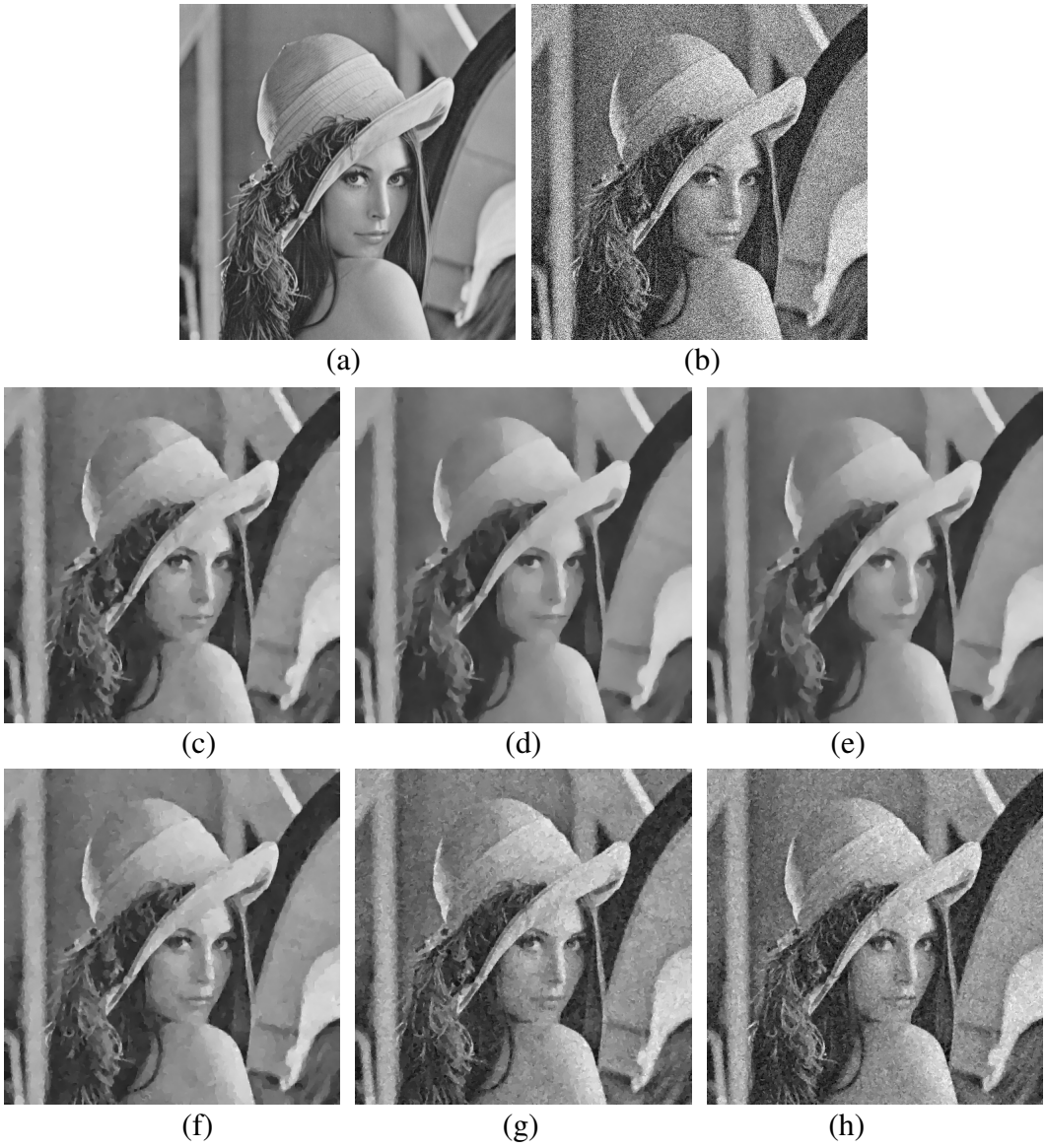


Figure 2.7: TV method for image denoising: (a) an original *Lena* image, (b) an AWGN noisy image with $\sigma = 20$, (c) a denoised image using a TV filter ($iteration = 50, \lambda = auto, \triangle t = 0.2$), (d) a denoised image using a TV filter ($iteration = auto(127), \lambda = auto, \triangle t = 0.2$), (e) a denoised image using a TV filter ($iteration = 200, \lambda = auto, \triangle t = 0.2$), (f) a denoised image using a TV filter ($iteration = auto(121), \lambda = 0.02, \triangle t = 0.2$), (g) a denoised image using a TV filter ($iteration = auto(122), \lambda = 0.03, \triangle t = 0.2$), and (h) a denoised image using a TV filter ($iteration = auto(108), \lambda = 0.04, \triangle t = 0.2$).

where *div* indicates the divergence operator, $\triangle$ is the Laplacian operator, ∇ is the gradient operator and t is a time scale. Equation 2.19 would be reduced to being an isotropic heat diffusion equation, if $c(x, y, t)$ were to be constant. It is worth mentioning that the smoothing amount is controlled by the value of $c(x, y, t)$. Perona and Malik attempted to automatically set the $c(x, y, t)$ values to be close to 0 to prevent any smoothing across boundaries and to be close to 1 in smoothing the interior regions. Equation 2.19 can be discretized on 4-nearest neighbours as

$$\begin{aligned} \hat{u}(x,y)^{t+1} &= \hat{u}(x,y)^{t} + \lambda\left[c_N(x,y)^{t}\cdot\nabla_{-}^{x}\upsilon(x,y)^{t} + c_S(x,y)^{t}\cdot\nabla_{+}^{x}\upsilon(x,y)^{t}\right. \\ &\quad \left. + c_E(x,y)^{t}\cdot\nabla_{+}^{y}\upsilon(x,y)^{t} + c_W(x,y)^{t}\cdot\nabla_{-}^{y}\upsilon(x,y)^{t}\right] \end{aligned} \tag{2.20}$$

where $0 \leq \lambda \leq 1/4$ is used for stability, ∇_{-}^{x}, ∇_{+}^{x}, ∇_{-}^{y} and ∇_{+}^{y} are the neighbour differences computed as Equation 2.16 , and C_N, C_S, C_E and C_W are the conduction coefficients of the four neighbours for each t iteration at the location (x, y). The conduction coefficients are defined as

$$\begin{aligned} C_N(x,y)^{t} &= g\left(\| \nabla_{-}^{x}\upsilon(x,y)^{t} \|\right) \\ C_S(x,y)^{t} &= g\left(\| \nabla_{+}^{x}\upsilon(x,y)^{t} \|\right) \\ C_E(x,y)^{t} &= g\left(\| \nabla_{+}^{y}\upsilon(x,y)^{t} \|\right) \\ C_W(x,y)^{t} &= g\left(\| \nabla_{-}^{y}\upsilon(x,y)^{t} \|\right) \end{aligned} \tag{2.21}$$

where $g(\cdot)$ function is defined as

$$g\left(\nabla I\right) = e^{-\left(\|\nabla I\|/K\right)^2} \tag{2.22}$$

or

$$g\left(\nabla I\right) = \frac{1}{1 + \left(\frac{\|\nabla I\|}{K}\right)^2} \tag{2.23}$$

where K is a constant that represents the noise standard deviation. $g(\cdot)$ function (Equation 2.22) is the Taylor approximation of $g(\cdot)$ function (Equation 2.23). Both functions perform similarly. The plot provided in Figure 2.8 illustrates the performance of the two functions when the inputs were 0-255. Figure 2.9 shows the effects of the number of iterations on the ADF when used for denoising an AWGN noisy `Lena` image. Increasing the number of iterations imparts more smoothing to the images. Figure 2.10 shows the effect of the parameter k on the ADF; here edges in the image remain sharp when the k value is low. The effect of the parameter λ on the ADF is shown in Figure

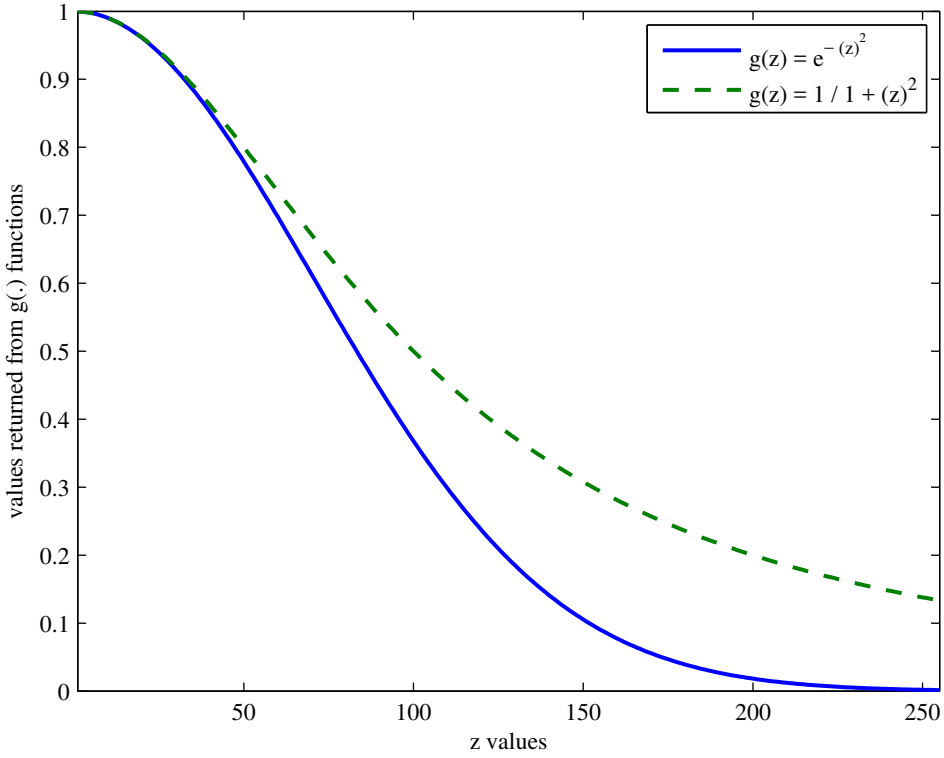


Figure 2.8: The performance of the $g(.)$ functions (Equation 2.22 and Equation 2.23) of the ADF.

2.11. When the λ value increases, more smoothing is applied to the image.

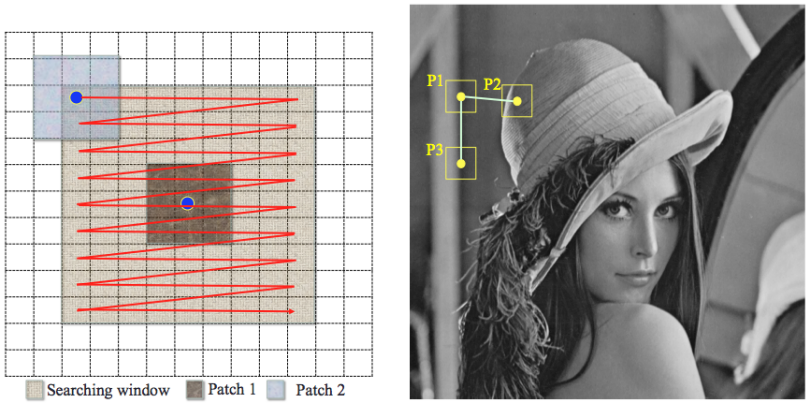
2.2.2 Patch-based Image Filtering

It is now common practice in image denoising to utilize patch-based models and algorithms instead of pixel-based approaches to produce most promising estimate of the noise-free images. However, there are advantages and disadvantages to the use of patch-based models and algorithms. Among the advantages, the smoothing of flat regions is the most important. Redundancy between patches enable patch-based approaches to properly smooth flat regions. Another advantage is the preservation of fine image details and sharp edges. Among the disadvantages, the similarity between patches assists in estimating flat region, and so is averaging, but it is quite time-consuming to group and compare similar patches. Therefore, each patch could have multiple estimates and patches are overlapped. Secondly, while it may be that patterns and textures are seemingly clear, patch-based models and algorithms usually exploit a large number of parameters, which can be extremely difficult to adjust properly.

Figure 2.9: The effects of a number of iterations on ADF: (a) an original *Lena* image, (b) an AWGN noisy image with $\sigma = 20$, (c) a denoised image using ADF (***iterations* = 50**, $\lambda = 0.25$, $k = 0.03$), (d) a denoised image using ADF (***iterations* = 100**, $\lambda = 0.25$, $k = 0.03$), (e) a denoised image using ADF (***iterations* = 150**, $\lambda = 0.25$, $k = 0.03$), and (f) a denoised image using ADF (***iterations* = 200**, $\lambda = 0.25$, $k = 0.03$).

Figure 2.10: The effects of K on ADF: (a) an original *Lena* image, (b) an AWGN noisy image with $\sigma = 20$, (c) a denoised image using ADF (*iterations* = 100, $\lambda = 0.25$, $\boldsymbol{k = 0.02}$), (d) a denoised image using ADF (*iterations* = 100, $\lambda = 0.25$, $\boldsymbol{k = 0.03}$), (e) a denoised image using ADF (*iterations* = 100, $\lambda = 0.25$, $\boldsymbol{k = 0.04}$), and (f) a denoised image using ADF (*iterations* = 100, $\lambda = 0.25$, $\boldsymbol{k = 0.05}$).

Figure 2.11: The effects of λ on ADF: (a) an original *Lena* image, (b) an AWGN noisy image with $\sigma = 20$, (c) a denoised image using ADF (*iterations* = 100, $\boldsymbol{\lambda = 0.10}$, $k = 0.03$), (d) a denoised image using ADF (*iterations* = 100, $\boldsymbol{\lambda = 0.15}$, $k = 0.03$), (e) a denoised image using ADF (*iterations* = 100, $\boldsymbol{\lambda = 0.20}$, $k = 0.03$), and (f) a denoised image using ADF (*iterations* = 100, $\boldsymbol{\lambda = 0.25}$, $k = 0.03$).



(a) (b)

Figure 2.12: Similarity between patches: (a) NL-Means patches as a raster scan in a search window, and (b) Patch P3 is more similar to P1 than to Patch 2; hence, P3 will receive a higher weight than the P2 weight.

Regardless, we believe that the advantages of patch-based methods far outweigh their disadvantages, as modern computers are significantly overpowered, and have large memory banks.

2.2.2.1 Averaging Patch-Based: Non-local Means

NL-Means is a patch-based filter proposed by Buades *et al.* [17] as a modification of the pixel-wise BF [95, 85, 13, 37, 68, 99]. Like the BF, the NL-Means filter blurs homogeneous areas and preserves edges. The NL-Means filter divides the input image into sub-images and then filters each sub-image separately in patch-wise fashion. Each sub-image contains several patches. As in the BF, similarity is measured based on two measurements: (1) the Euclidean distance between the centres of the patches, and (2) the luminance distance between the patches. In contrast to the BF, patches are compared within a searching window instead of with the pixels of its neighbours. Thus, it is called a non-local method. Patches with similar grey scale levels have larger weights when they are averaged. Figure 2.12 (a) shows the NL-Means patches and how to find similar patches in a raster scan search window. Figure 2.12 (b) illustrates that patches with a similar grey scale level, for example, P1 and P3, should be assigned a higher weight over those assigned to P1 and P2. Figure 2.13 shows the weight range from 1 (white) to zero (black) for the displayed image; 1 indicates that there are two identical patches, and vice versa. The edges in NL-Means filtering are preserved regardless of their direction.

The estimated value $NL[v]_i$, for a pixel i, is computed as in Equation 2.24,

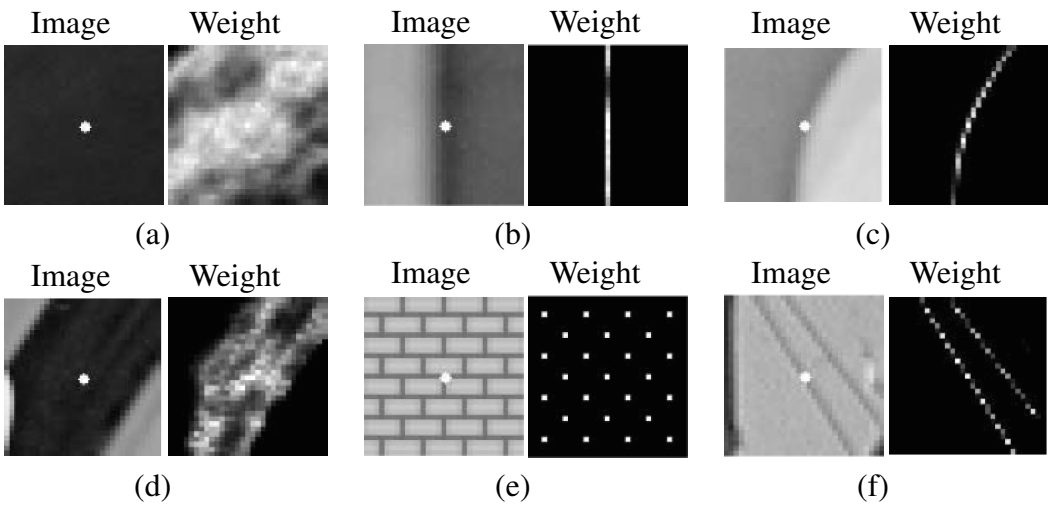


Figure 2.13: The weights distribution used to estimate the central pixel of each searching window in NL-Means. The weights range from 1 (white) to zero (black) [17].

$$NL\,[v]_i = \sum_{j \in I} \omega(i,j)\,[v]_j \tag{2.24}$$

where $[v]_i$ and $[v]_j$ are pixel intensities at location i and j, respectively, and $\omega(i,j)$ is a similarity measure between the pixels i and j. The similarity measure of weight satisfies the condition $0 \leq \omega(i,j) \leq 1$ and $\sum_j \omega(i,j) = 1$. The similarity weight depends on the grey scale level's similarity and the Euclidean distance between vectors $N\,[v]_i$ and $N\,[v]_j$, where $N\,[v]_k$ denotes a square neighbourhood of a fixed size and centred at a pixel k. The weights are described as,

$$\omega(i,j) = \frac{1}{Z(i)} e^{-\frac{\|(N[v]_i)-(N[v]_j)\|^2}{h^2}} \tag{2.25}$$

where $Z(i)$ is a normalization factor and h is a filtering parameter set depending on the noise level.

The level of noise determines the size required for the search windows and patches. For a robust comparison between patches, the size of the patches increases when the noise level is high. Accordingly, the value of the filtering parameter, h increases as the size of the patch is increased. Meanwhile, the size of the search windows must be increased in order to find more similar patches.

The NL-Means filter has many modifications. Improving the weight assignment between the patches improves the performance of the NL-Means method. Hedjam *et al.* [48] improved the process of adjusting the weights in the NL-Means by using Markovian clustering. Wu *et al.* [101] applied a statistical shrinkage perspective when assigning the weights in NL-Means using

a James-Stein [56] shrinkage estimator. Lai *et al.* [63] introduced an improved neighbourhood pre-classification strategy for optimizing the weight kernels of NL-Means filter. Khan *et al.* [60] introduced a variant of the NL-Means scheme by using a thresholding step to reduce the number of similar patches before the weighted averaging of the patches.

The NL-Means filter exists in the spatial domain. Applying this filter in the frequency domain would assist in suppressing a noisier signal. Alfredo *et al.* [38] transferred the patches of the NL-Means to the frequency domain, and used *discrete cosine transform* (DWT) with a set threshold to estimate true patches. Zhong *et al.* [109] combined the NL-Means with a Lee filter [65] for SAR images enhancing. Chan *et al.* [18] incorporated a median filtering operation indirectly in the NL-Means method for denoising low *signal noise ratio* (SNR) images. Maruf *et al.* [75] projected NL-Means patches into a global feature space before performing a statistical *t*-test to reduce the dimensionality of this feature space. Irrera *et al.* **[54]** adapted NL-Means for denoising X-ray images (XNLM), and then applied an additional multiscale contrast enhancement stage in the frequency domain.

An NL-Means filter could be adapted to improve other image processing applications (e.g. segmentation, recognition, and video denoising). Zhan *et al.* [105] introduced an extension to the NL-Means method for ultrasonic speckle reduction. They assigned the patches' similarity weights iteratively in a lower dimensional subspace using *principal component analysis* (PCA). Xu *et al.* [103] adapted the NL-Means for microscopy cell images using a frequency transform. Genin *et al.* [42] adapted a modified version of the NL-Means filter for detecting small objects by background suppression. Background pixels were estimated by applying a weighted average depending on the similarity between the neighbouring pixels. Kim *et al.* [61] adapted the NL-Means filter for noise reduction and the enhancement of extremely low-light video. They use a motion adaptive temporal filter using Gamma correction with adaptive thresholds before using the NL-Means filter. Xu *et al.* [102] adapted the idea of patching from the NL-Means for filtering *polarimetric synthetic aperture radar* (POL-SAR) images; they used simultaneous sparse coding for the transferring of the patches into the frequency domain before assigning the weights.

2.2.2.2 Probabilistic Patch-Based Filtering

The *probabilistic patch-based filter* (PPB) of Deledalle *et al.* [32], which works in the spatial domain, is an extension of the NL-Means filter. The PPB approach is one of the few denoising techniques that can provide a general denoising methodology for various noise models. Thus, it is more general than the NL-Means, and can be applied where there is either additive noise or multi-

plicative speckle noise. The PPB filter is a statistically-based similarity scheme that depends on the distribution model of the noise. The weighted average is used for the Gaussian noise distribution in the NL-Means, but the PPB filter applies smoothing based on the *maximum likelihood estimator* (MLE). The PPB is expressed as a *weighted maximum likelihood estimation* (WMLE) problem. The weight is derived from the data by improving the isotropy of the filter - non-*iterative PPB filter* (Non-iPPB), and it can be iteratively defined based on the similarity of the patches - *iterative PPB filter* (It-PPB).

Weighted Maximum Likelihood Estimator: Using *weighted* MLE for image denoising is not a new technique; it was first used for image denoising by Polzehl and Spokoiny [88, 87]. The PPB filter redefined the weights in terms of a patch-based approach. This form of image denoising is considered to be an estimation $\hat{u}$ of the true image u which originates from the noisy image v. The images are defined over a discrete regular grid Ω. A pixel value is described as i, and its neighbour is j at the location $(x, y) \in \Omega$. The noise model is considered as being defined by the parametric noise distribution "likelihood" $p(i \mid \theta_i^*)$, where θ_i^* is an unknown space-varying parameter. The denoising of an image is equivalent to finding the best estimation $\hat{\theta}_i$ for θ_i^* for all of the pixels. The MLE at each location (x, y) estimates $\hat{\theta}_i$ from a set $S_{\theta_i^*}$ of the distributed random variables around it by

$$\begin{aligned} \hat{\theta}_i &\triangleq \arg\max_{\theta_i} \sum_{j \in S_{\theta_i^*}} \log p\left(j \mid \theta_i\right) \\ &\triangleq \arg\max_{\theta_i} \sum_{j} \delta_{S_{\theta_i^*}}(j) \log p\left(j \mid \theta_i\right) \end{aligned} \tag{2.26}$$

$\delta_{S_{\theta_i^*}}$ is an indicator function for $S_{\theta_i^*}$ (i.e., $\delta_{S_{\theta_i^*}} = 1$ if $j \in S_{\theta_i^*}$, or 0 otherwise). The indicator function has been derived from the data as weights $\omega(i, j) \geq 0$ in [32, 87, 88], where it is used as the weight for adaptive pixel-wise filters. However, the indicator function in PPB is used as a weight function to form the WMLE:

$$\hat{\theta}_i \triangleq \arg\max_{\theta_i} \sum_{j} \omega(i, j) \log p(j \mid \theta_i) \tag{2.27}$$

Defining the Weight Between Patches: In Subsection 2.2.2.1 on page 26, the weights in the NL-Means filter are defined by comparing the similarities of the two patches $[v]_i$ and $[v]_j$ which are

centred around the two locations i and j, respectively. A weighted Euclidian distance between the two patches defines the level of similarity. However, the objective of the PPB filter is to generalize and extend the idea of the Euclidean distance weight used in the NL-Means filter so that it can be adapted to non-additive noise models. The weights used in the PPB filtering method are estimated using the probability of the two patches in a noisy image having the same parameters. By following the same idea as the weight in the NL-Means filter and assuming equal values for i and j in the two statically similar patches $[v]_i$ and $[v]_j$, the PPB weights would be defined as

$$\omega(i,j)^{(PPB)} \triangleq p\left(\theta^*_{[v]_i} = \theta^*_{[v]_j} \mid \upsilon\right)^{1/h} \tag{2.28}$$

where $\theta^*_{[v]_i}$ and $\theta^*_{[v]_j}$ are the patches extracted from the image θ^*, and h is larger than 0 , which indicates the size of the patch in PPB. The h acts as σ in the NL-Means algorithm in order to control the filtering amount. The probability of the similarity of the patches pixels is decomposed into a product of the probabilities of k neighbours: $\prod_k p\left(\theta^*_{i,k} = \theta^*_{j,k} \mid \upsilon_{i,k}, \upsilon_{j,k}\right)$.

Iterative (WMLE) Denoising: In order to improve the performance of the PPB algorithm, the probability of a similarity is estimated iteratively. The weight, at each iteration, is expressed as the product of two terms: (1) the probability of the similarity between the noisy patches as was described in Defining the Weight Between Patches's section on page 29, and (2) the probability of the similarity derived from the previous iteration. Assume that the previous estimation at t iteration is $\hat{\theta}^{t-1}$ for θ^*. Then, the formula in Equation 2.28 can be expressed as:

$$\omega(i,j)^{(It.PPB)} \triangleq p\left(\theta^*_{[v]_i} = \theta^*_{[v]_j} \mid \upsilon, \hat{\theta}^{t-1}\right)^{1/h} \tag{2.29}$$

This is similar to what was achieved in Defining the Weight Between Patches's section on page 29, where the probability of the similarity is decomposed into a product of the probabilities of the k neighbours: $\prod_k p\left(\theta^*_{i,k} = \theta^*_{j,k} \mid \upsilon_{i,k}, \upsilon_{j,k}, \hat{\theta}^{t-1}\right)$. From the Bayesian framework, the naïve Bayes model can be fitted with the maximum likelihood concept. The probability is estimated using the prior probability, and can be presumed to be proportional to the likelihood $p\left(\upsilon_{i,k}, \upsilon_{j,k}, \hat{\theta}^{t-1} \mid \theta^*_{i,k} = \theta^*_{j,k}\right)$. The similarity likelihood is computed using:

$$p\left(\theta^*_{i,k} = \theta^*_{j,k} \mid \upsilon_{i,k}, \upsilon_{j,k}, \hat{\theta}^{t-1}\right) \propto \underbrace{p\left(\upsilon_{i,k}, \upsilon_{j,k} \mid \theta^*_{i,k} = \theta^*_{j,k}\right)}_{likelihood} \times \underbrace{p\left(\theta^*_{i,k} = \theta^*_{j,k} \mid \hat{\theta}^{t-1}\right)}_{prior} \tag{2.30}$$

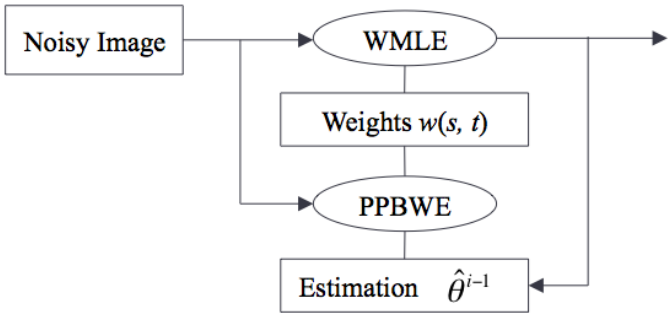


Figure 2.14: Competing iteratively the weights between the two pixels s and t in the *probabilistic patch-based filter*. The PPB Weights Estimator (PPBWE) uses the noisy image and the estimated values from the previous iteration in order to estimate the weight.

The likelihood term computes the degree of similarity between the patches, and the prior term compares the two probability distributions from the previous iteration, similar to [87].

The scheme, Figure 2.14, shows the procedure of iteratively competing with the weights in the PPB algorithm. The procedure for defining the weights is estimated iteratively by the following methods: (1) the *PPB weights estimator* (PPBWE) uses the likelihood term and the estimated value from the previous iteration in order to compute the prior term (Equation 2.30), (2) the WMLE uses the PPBWE estimation and the noisy image to estimate the new weight (Equation 2.27), and (3) the PPBWE and the WMLE steps are repeated until there is no difference in the estimations made in the two steps.

Algorithm Used in the Case of Gaussian Noise: By assuming the AWGN model, the values of the pixels I of the patch $[v]_i$ are distributed based on the Gaussian distribution $\aleph\left(u, \sigma^2\right)$. Here, u is the noiseless image and σ the noise variance. The noiseless image u can be estimated by the weighted average that maximizes the WMLE defined in Equation 2.27:

$$\ddot{u}_i^{(WMLE)} = \frac{\sum_j \omega(i,j)\, I_j^2}{\sum_j \omega(i,j)} \tag{2.31}$$

In order to estimate the weighted average $\omega(i,j)$, the likelihood and the prior terms are considered. The likelihood function is discretizing as:

$$\underbrace{p\left(I_{i,k}, I_{j,k} \mid \ddot{u}_{i,k} = \ddot{u}_{j,k}\right)}_{likelihood} \propto \exp\left(-\frac{|\, I_i - I_j\,|^2}{4\sigma^2}\right) \tag{2.32}$$

, and the prior term is discretized as:

$$\underbrace{p\left(\ddot{u}_{i,k} = \ddot{u}_{j,k} \mid \ddot{u}^{t-1}\right)}_{prior} \propto \exp\left(-\frac{1}{T}\frac{\mid \ddot{u}^{t-1}_{i,k} - \ddot{u}^{t-1}_{j,k} \mid^2}{\sigma^2}\right) \tag{2.33}$$

By combining the two terms in Equation 2.32 and 2.33, the weight at any iteration is defined as:

$$\omega(i,j)^{(It.PPB)} = \exp\left[-\sum_n \left(\frac{1}{h}\frac{\mid I_i - I_j \mid^2}{4\sigma^2} + \frac{1}{T}\frac{\mid \ddot{u}^{t-1}_{i,k} - \ddot{u}^{t-1}_{j,k} \mid^2}{\sigma^2}\right)\right] \tag{2.34}$$

where n is the number of pixels and T is a constant similar to h in Equation 2.25. When there is no iteration "*posterior* term = 0", the filter performs in a similar way to the NL-Means filter.

A further *extend patch log likelihood* (EPLL) filter, similar to PPB filter, was proposed recently by Papyan *et al.* [82] using a multi-scale prior.

2.2.2.3 Dictionary Learning

Dictionary learning (DL) is used as a replacement for the use of a fixed dictionary to represent data. Since the seventies, data can be represented by using a fixed dictionary; for instance, Fourier [15] and Wavelets [45]. Fifteen years ago, Olshausen *et al.* proposed an approach to learning the dictionary from a data set in order to optimize the sparsity of the data [80]. The DL or *k-means singular value decomposition* (K-SVD) was first adapted to image denoising in 2006 by Aharon *et al.* [5]. The DL method finds the best dictionary $D = (d_i)_{i=1}^z$ of z atoms $d_i \in \mathbb{R}^n$ that sparses the a set $Y = \left(y_j\right)_{j=1}^m \in \mathbb{R}^{n\times m}$ of signals $y_j \in \mathbb{R}^m$. In order to filter a noisy image, each signal y_j is considered as a patch extracted from a noisy image. The sparse code of signal data $y = y_j$ for $j = 1, \ldots, n$ is obtained by minimizing a constrained optimization ℓ^0:

$$\min_{\|x\|_0 \leq k} = \frac{1}{2}\|y - Dx\|^2 \tag{2.35}$$

where $k > 0$ controls the amount of sparsity, and ℓ^0 pseudo-norm is defined by:

$$\|x\|_o = \{i : x_i \neq 0\} \tag{2.36}$$

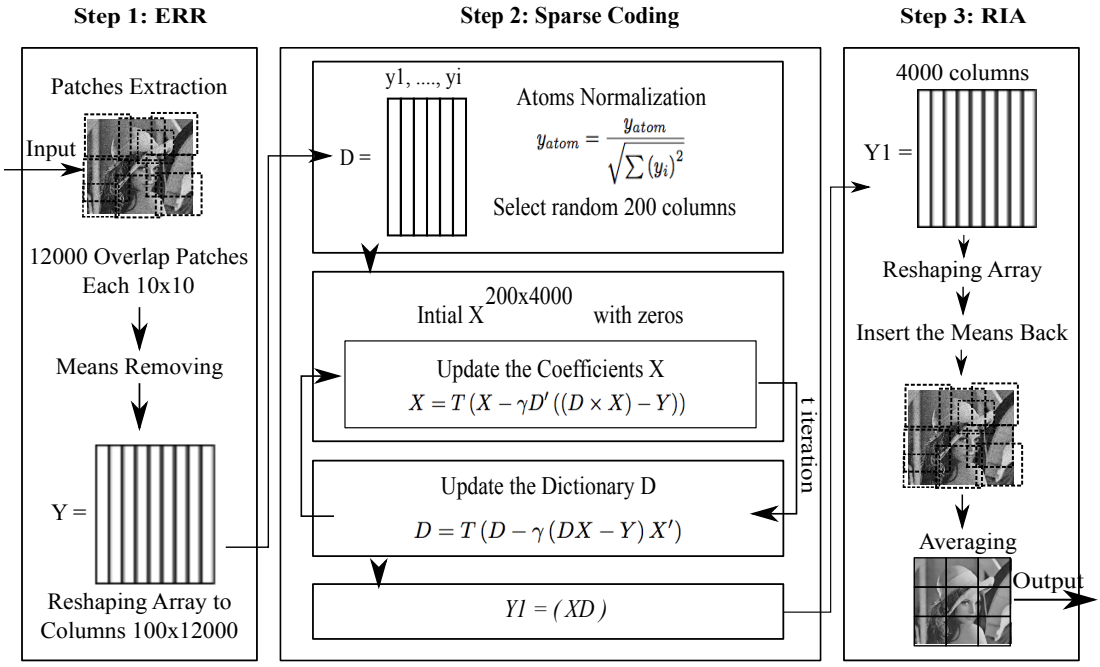


Figure 2.15: *Dictionary learning* filtering scheme

In DL, optimization is performed on the dictionary D and the coefficients $X = \left(x_j\right)_{j=1}^{m} \in \mathbb{R}^{p \times m}$ for $j = 1, \ldots, n$, where the set of coefficients is x_j of the data y_j. The joint optimization is written as:

$$\underset{D \in \Phi, X \in \chi_k}{\arg\min} E(X, D) = \frac{1}{2} \|Y - DX\|^2 = \frac{1}{2} \sum_{j=1}^{m} \left\|y_j - Dx_j\right\|^2 \tag{2.37}$$

where Φ is the constraint set:

$$\Phi = \{D \in \mathbb{R}^{n \times p} : \forall i \ \|D., i\| \leq 1\} \tag{2.38}$$

The sparsity constraint is set on χ_k , which is the unit normalization of the dictionary columns:

$$\chi_k = \{X \in \mathbb{R}^{p \times m} : \forall i \ \|X., i\|_0 \leq k\}. \tag{2.39}$$

Peyré *et al.* [84] proposed using the block-coordinate descent minimization approach used by Tseng [96] in order to minimize X and D. The scheme in Figure 2.15 shows the steps of the DL method for denoising images.

The DL algorithm depends mainly on three steps: (1) patch extraction, (2) sparse coding, and (3) patch construction. In the first step, the patches are randomly extracted from the whole input image. In the sparse coding step, the energies of the X and D dictionaries are iteratively minimized. Patches averaging and reconstruction occur in the patches reconstruction step. The algorithm of DL is shown in Algorithm 2.1.

Algorithm 2.1 *Dictionary learning* filtering method

1. Patch Extraction Step:
   (a) The mean of each patch is removed from each pixels value.
   (b) Patches are sorted based on their energy; the patches with a high level of energy are kept by thresholding.
   (c) Patches are reshaped as columns in order to form Y.
2. Sparse Coding Step:
   (a) In the columns each atom is normalized in order to form the initial dictionary D.
   (b) The number of the columns is reduced again, before computing the X coefficients.
   (c) The X dictionary is initially started with zeros.
   (d) The coefficients of dictionary X is updated by: $X = T\left(X - \gamma D'\left((D - X) - Y\right)\right)$. Where T is the threshold.
   (e) The dictionary D is updated again by: $D = T\left(D - \gamma\left((DX - X) - X'\right)\right)$.
   (f) K is the number of iterations used to minimize the X and D dictionaries by updating them iteratively.
   (g) Finally, the coefficients of dictionary X are multiplied by dictionary D.
3. Patch Construction Step:
   (a) The result of the multiplication is a new array, $Y1$.
   (b) The columns of $Y1$ are reshaped to form the patches.
   (c) Re-inserting the averages into the patches precedes the averaging of the patches to replace the noisy patch.

The DL method has many modifications. Tian *et al.* [94] made the DL more sparsely representative in the case of fewer observation values by proposing an adaptive orthogonal matching pursuit to adaptively ensure the sample size. Some of the modifications aim to adapt the idea of denoising based on using DL in other image processing applications. Chen *et al.* [23] generalized the idea of

the learning dictionary to explore identity information in multiple frames of videos. They generated a sparse representation from multiple video frames for face and body part recognition. Fu *et al.* [41] proposed an effective model based on DL for *hyper-spectral image* (HSI) denoising via considering the issue of sparsity across the spatial-spectral domain, high correlation across spectra, and non-local self-similarity over space. Kang *et al.* [58] proposed a feature-based approach for assessing similarity between images. After extracting feature points from an image, they utilized DL. They then measured the similarity between images in terms of sparse representation. A novel self-learning based image decomposition framework was presented by Huang *et al.* [52]. Their framework performs unsupervised clustering on the observed dictionary via affinity propagation, to identify image components with similar context information. The framework can automatically determine the undesirable random noisy components from true image components directly from a noisy image. DL algorithm was adapted to filter Chinese character images by Zhenghao *et al.* [91]. They divided the image frequency to low and high frequency. While butterworth low-pass filter was utilized to filter low frequency, DL algorithm was proposed to filter high frequency which consists of Chinese character structure.

2.2.2.4 Patch-Based PCA

Recently, over-complete dictionaries with sparse representation techniques became very widespread in image denoising [5, 72, 71]. These methods use over-complete dictionaries derived from enormous image sets or from the noisy image itself. They outperform other denoising techniques due to their ability to provide an appropriate basis for separating noisy signals from the true image signals. Despite the fact that over-complete dictionaries are frequently used for image denoising; such dictionaries are sophisticated and quite expensive in terms of memory usage and time. However, *patch-based principal component analysis* (PB-PCA) of Deledalle *et al.* [22] is a modification of the dictionaries' methods.

PB-PCA uses simple orthogonal dictionaries, which are constructed by using PCA. The results of PB-PCA still do not outperform the over-complete dictionary methods, but PB-PCA shows how simple orthogonal dictionaries can achieve excellent results with less sophistication. This form of analysis simply learns the orthogonal dictionaries from the noisy image via PCA. The next step is to threshold the patches' coefficients in the dictionaries. This idea is similar to the wavelet denoising methods used in [36, 20, 21], in which they use either hard-thresholding or soft-thresholding for zeroing the coefficients. Figure 2.16 shows the extraction of the patches used in the PB-PCA method from an image and grouping them before computing the PCA.

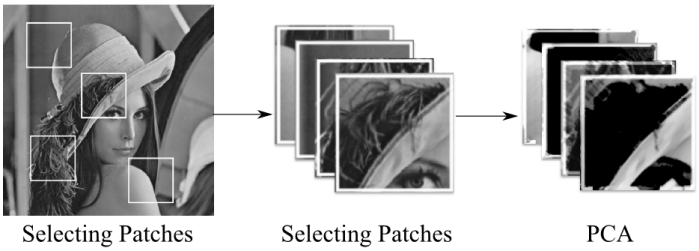


Figure 2.16: Extracting patches in the PB-PCA method and grouping them before PCA

When denoising an image with AWGN, the patch model has the following formula:

$$[v]_i = [u]_i + z_i, i = 1, .., n-1 \tag{2.40}$$

where $[u]_i$ is the true image patch, z_i is the AWGN noise, $[v]_i$ is the noisy patch and n is the number of patches. The patches are overlapped. By assuming $[v]_i, .., [v]_{i-1}$ are a group of patches of size $N \times N$ extracted from the noisy image υ, the covariance matrix is the sum of:

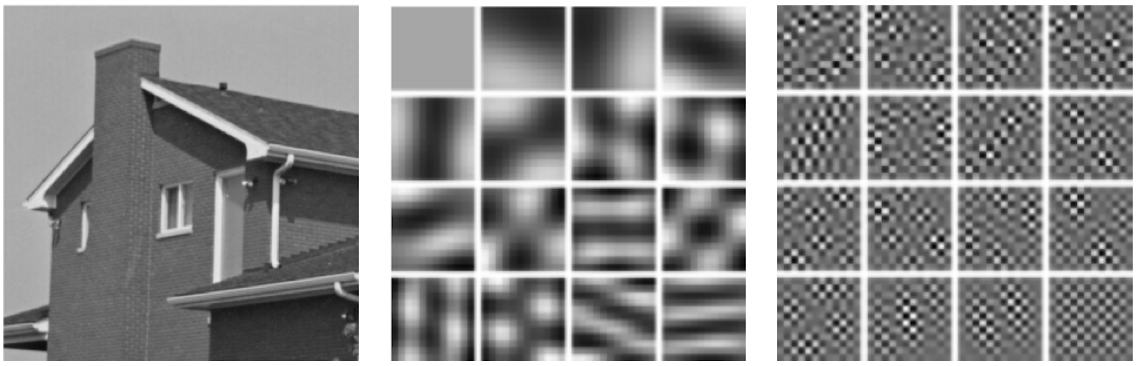
$$\sum = \frac{1}{n} \sum_{i=1}^{n} [v]_i [v]_i' - \bar{\upsilon}\bar{\upsilon}' \tag{2.41}$$

where

$$\bar{\upsilon} = \frac{1}{n} \sum_{i=1}^{n} [v]_i$$

In PCA, the *singular value decomposition* (SVD) of the covariance matrix $\sum$ is processed. Moreover, the eigenvalues $g_1, \cdots, g_{n-1}$ of the covariance matrix and the corresponding eigenvectors $G_1, \cdots, G_{n-1}$ are calculated. Eigenvectors are called the principal components "axis" of the processed data and are used to form an orthogonal basis, G_i is the the i^{th} principal axis of the data. Due to the orthogonal basis of the principal components, an image patch can be decomposed as $[v]_i = \sum_{i=1}^{n} \langle [v]_i \mid G_i \rangle G_i$. Figure 2.17 (b) and (c) show the first and last 16 principal axes of the all the patches obtained from the house image shown in Figure 2.17 (a).

By assuming that the true image pixels have a low dimensional subspace, and the noise is spread in all directions, projecting the axes into the first axis would suppress the noise in the noisy image. Projecting the axes is called *coefficient thresholding*, and it is done by using an appropriate shrinkage function. A general formula for estimating a true image is:



(a) (b) (c)

Figure 2.17: The principal components "axis" of the house image: (a) is the input image, (b) is the first 16 principal axes of the all the patches obtained from the house image, and (c) is the last 16 principal axes of the all patches obtained from the house image. [22]

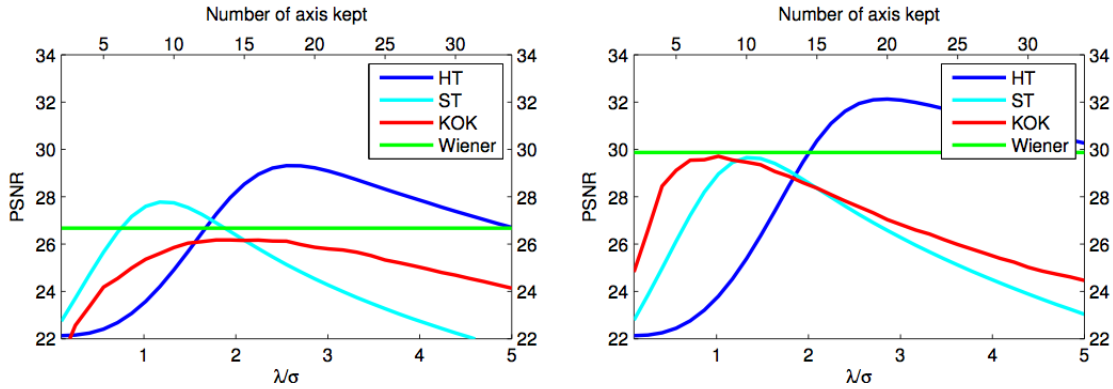


Figure 2.18: A comparison between PSNR of the different methods of the projections in PB-PCA for the House and Cameraman images. The threshold ratio (λ/σ) into the bottom of the x-axis controls the number of axes kept in the upper x-axis. Here, σ is the noise variation and λ is chosen by cross validation [22]

$$[\hat{u}]_i = \bar{v} + \sum_{i=1}^{n} \eta\left(\langle [v]_i - \bar{v} \mid G_i \rangle\right) G_i \tag{2.42}$$

where η is the shrinkage function.

The PB-PCA filtering method has been tested with four shrinkage functions: (1) Soft Thresholding (ST); (2) *hard thresholding* (HT); (3) *keep or kill* (KoK), and (4) Wiener filter [100]. Figure 2.18 shows a comparison between the four different projection methods of the PCA basis for the `House` and `Cameraman` images. From Figure 2.18, it can be seen that hard-thresholding with a large number of axes is the best of the four projection methods. Using a Winer filter as a shrinkage function for the PCA has been proposed by Zhang *et al.* and Muresan *et al.* in [106, 77].

(a) (b) (c)

Figure 2.19: Variants of collecting a set of patches before PCA process in PB-PCA filtering: (a) Global PCA, (b) Local PCA, and (c) Hierarchical PCA.

The PB-PCA method has three variants based on how each set of patches is collected from the input noisy image before the PCA process. In PB-PCA, the three variants for collecting patches are: globally, locally or hierarchically. Figure 2.19 shows the best means of collecting the patch sets globally, locally or hierarchically.

Patch-Based Global PCA: In the *patch-based global PCA* (PGPCA) approach, collecting patches for a PCA can be done globally from the entire noisy image. This approach has been used by Muti *et al.,* and Bacchelli *et al.* [78, 11, 106]. The Bacchelli *et al.* [11] algorithm uses a linear transformation, "a wavelet transform", before computing the PCA in order to achieve better results. The Zhang *et al.* [106] algorithm collects patches globally, but it computes the PCA in two stages. The PGPCA approach is faster than the other local or semi-local approaches, which need time to divide the image into sub-images before computing the PCA coefficients. In terms of the filtering quality, the PGPCA approach cannot compete with the other approaches that consider the high level of redundancy occurring between neighbouring patches. One original basis for the whole image is utilized in the global PGPCA, which impacts negatively on the denoising process. The global PGPCA does not identify the rare patches because they do not exert a strong influence on the total variance. However, an allowance can be made for these limitations by considering the local redundancy between patches.

Patch-Based Local PCA: In the *patch-based local PCA* (PLPCA) approach, patches are collected locally in order to overcome the limitations of the global PGPCA approach. The local collection of patches means that the patches are collected within a small region of interest in the noisy image. A fixed search window $N \times N$ is applied to the whole image. Since the patches are

overlapped in PLPCA, there will be multiple estimates for a single pixel. Averaging is used to compute a single pixel's value.

The advantage of this approach is that the orthonormal basis is adapted only to the sub-image not to the whole image. However, this approach has two limitations: the overfitting and the fact that it is time consuming. The overfitting problem is due to the limited number of patches on which to compute the PCA. PLPCA is extremely time-consuming because the PCA needs to be computed repeatedly.

Fei *et al.* [39] collected patches locally, but they improved their approach by using geometric structure clustering to guarantee that only patches with similar properties were gathered. Pal *et al.* [81] considered patch redundancy in order to improve on the global two-stage PCA approach of Zhang *et al.* [106]. A sliding window that moves with a step $s = \frac{W_p - 1}{2}$, where s is the step size and W_p is the window's current location was utilized as a modification in order to reduce the time-consuming element of the PLPCA approach. The computational time is divided by s^2 without losing the denoising quality. Zhang *et al.* **[108]** proposed using similar *patch-based local PCA* filter with an extended step where a Wiener filter is finally applied.

Patch-Based Hierarchical PCA: In the *patch-based hierarchical PCA* (PHPCA) approach, an algorithm builds a hierarchical cluster of the patches. Clustering is the task of grouping a set of patches into the same cluster, i.e., a set. There are different cluster models; each model has several clustering algorithms. The models include: connectivity models (e.g. hierarchical clustering), and centroid models (e.g., k-means). The hierarchical clustering is based on the concept of grouping of patches according to a maximum distance between patches. Patches are represented as a dendrogram, which is a Greek word meaning a tree diagram that illustrates the arrangement of the patches. In centroid models, vectors are usually assigned to a number k of fixed clusters. For more information about the clustering models, readers are referred to Chapter 17 in "*Introduction to Information Retrieval*," by Manning *et al.* [74].

The objective of the PHPCA approach is to offer a solution that can provide a balance between the PGPCA and PLPCA approaches, which is less time-consuming than the local approaches and is more adaptable to local sub-images. The PHPCA approach uses a geometric partitioning to first divide the image into four areas, and then it estimates the principal axis for each area. Each area has its own principal components. This process is repeated until the end of the tree is reached. Several dictionaries share the first axes. Figure 2.15 shows how an image can be divided into sub-images.

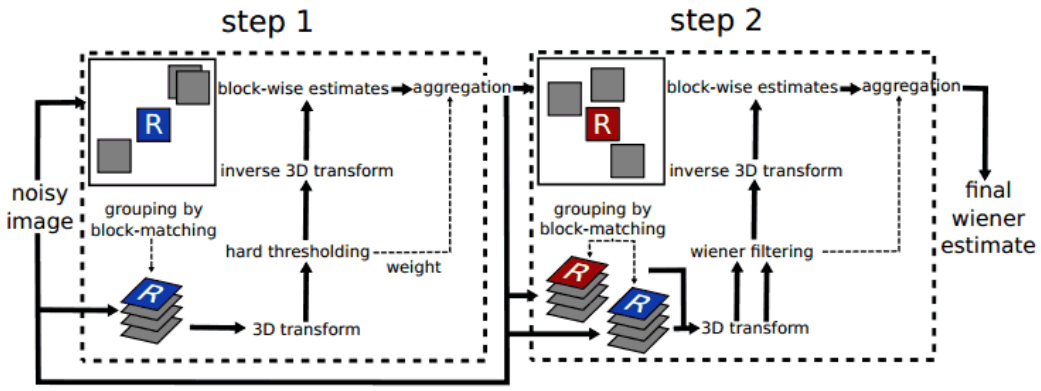


Figure 2.20: The two steps of BM3D filtering [24]

2.2.2.5 Block Matching 3D Filtering

BM3D filter is the state-of-the-art denoising technique by Dabov *et al.* [24]. It is based on modified sparse representation in the transform domain. The BM3D technique first groups the patches into 3D data arrays instead of into 2D arrays, and then it applies a modified sparse representation in the transform domain. Collaborative filtering is used to deal with the 3D arrays. The BM3D algorithm depends on two steps: (1) collaborative hard thresholding and (2) the collaborative Wiener filtering. The two steps allow the BM3D algorithm to suppress more noise and to preserve more details. The amount of noise is suppressed in the thresholding step, and the details are restored in the second step. Collaborative hard thresholding has three functions: (1) 3D transformation, (2) shrinkage and (3) 3D inverse transformation. The patches in the 3D arrays are overlapped, so a weighted average is used to obtain a single estimation for each pixel. Aggregation is the averaging procedure. A significant filter is obtained by using the BM3D algorithm. The scheme in Figure 2.20 shows the two steps of BM3D filtering. Below, the two steps of the BM3D algorithm are described. First, the collaborative hard thresholding step is explained. Second, the use of the collaborative Wiener filtering is discussed.

- **Step One: Thresholding**

Grouping: Similar to the NL-Means, a search window is used here to determine the similarity between the patches. The search window is used in order to benefit from the high redundancy among the neighbouring patches. There are several different grouping techniques, and a number of these techniques have already been discussed. Other useful techniques for grouping could be considered for patch grouping; such as, vector quantization [43], *k*-means clustering [70], self-organizing maps [62] and others [55]. However, grouping in BM3D is based on the similarity

distance between patches - the "Euclidian distance". The grouping stage is the first stage in both of the two steps' stages in which similar patches are gathered to form 3D arrays. Similarity is computed according to the similarity distances between the patches. In patches where the similarity distances are below a fixed threshold are considered to be similar and are grouped into the 3D array. Before measuring the distance, a coarse pre-filtering is used to linearly transform the patches using a 2D linear transformation such as: multiple wavelet transforms [89, 33]. The formulation in Equation 2.43 is used to compute the similarity distance between patches,

$$Dst\left([v]_i, [v]_j\right) = \frac{\left\|\gamma^{2D}\left(T^{2D}_{hard}\left([v]_i\right)\right) - \gamma^{2D}\left(T^{2D}_{hard}\left([v]_j\right)\right)\right\|^2_2}{\left(N^{hard}_1\right)^2} \tag{2.43}$$

where $[v]_i, [v]_j$ are, respectively, the reference patches at i and its neighbours at j, T^{2D}_{hard} is the 2D linear transform, γ^{2D} is a hard-thresholding operator equal to $\lambda_{2D} \times \sigma$ and $\left(N^{hard}_1\right)^2$ is the patch size $N \times N$. λ_{2D} is defined in Table 2.1 on page 44. σ is the estimated noise standard deviation. The 2D thresholding operator γ^{2D} makes all coefficients with absolute value less than the threshold ($\lambda_{2D} \times \sigma$) equal to zero, and it leaves the other coefficients unchanged. After computing the Euclidian distance, grouping the similar patches into a 3D array is required. The formulation in Equation 2.44 is used for gathering similar patches.

$$3D\,S^{hard}_i = \left\{j \in \Omega : Dst\left([v]_i, [v]_j\right) \leq T^{hard}_{match}\right\} \tag{2.44}$$

where $3D\,S^{hard}_i$ is the constructed 3D array contains similar patches, and T^{hard}_{match} is the maximum distance between two similar patches. The maximum grouped patches number is restricted to N^{hard}_2. The next stage is to apply the collaborative filter by: (1) performing a 2D linear transformation then a 1D linear transformation, (2) shrinkage, and (3) inverting the 1D transformation and the 2D linear transformation.

Collaborative Filtering: Once the 3D array is built, a collaborative filter is used for suppressing the noise. A 3D transformation is applied to the 3D array, before the shrinkage of the transforming coefficients. The 2D transformation in the Grouping stage is applied along both the horizontal and vertical lines for each patch, and then a third transformation is conducted along the third diminution of the 3D array for the 3D transformation. The formulation of the collaborative filter is:

$$3D\,\hat{u}_{S^{hard}_i} = T^{3D^{-1}}_{hard}\left(\gamma^{3D}\left(T^{3D}_{hard}\left(3D\,S^{hard}_i\right)\right)\right) \tag{2.45}$$

where T^{3D}_{hard} is the 3D linear transformation of the first (hard) step, $T^{3D^{-1}}_{hard}$ is the inverse of 3D transformation, and γ^{3D} is a hard-thresholding operator equal to $\lambda_{3D} \times \sigma$. λ_{3D} is defined in Table 2.1 on page 44. The 3D thresholding operator γ^{3D} makes all of the coefficients with absolute value less than the threshold ($\lambda_{3D} \times \sigma$) equal to zero.

Aggregation Weights: At this stage, the overlapped patches in the 3D array $\left(3D\,\hat{u}_{S_i^{hard}}\right)$ have multiple estimates for each pixel in the reference patch at a particular location i. A weighted averaging procedure is required to produce an estimate for each pixel. Weights in BM3D are inversely proportional to the total variance of the patches in the $3D\,\hat{u}_{S_i^{hard}}$ array. When the total variance is high, a small weight is assigned to the patch.

The amount of the additive noise is independent when processing the collaborative filter in Step One and Step Two. Thus, the total variance is not the same after applying the collaborative filter in the first and second steps. In Step One, the total variance is computed by $\sigma^2 \times N^{hard}_{non-zero}$, where $N^{hard}_{non-zero}$ is the number of non-zero coefficients after the hard-thresholding. The total variance calculated in Step Two depends on the results of the Wiener filter coefficients; the total variance of the Wiener filter will be explained on the following page. However, the weights for Step One are equal to:

$$\omega_i^{hard} = \begin{cases} \frac{1}{\sigma^2 \times N^{hard}_{non-zero}}, & \text{if} \rightarrow N^{hard}_{non-zero} \geq 1 \\ 1, & otherwise \end{cases} \tag{2.46}$$

- **Step Two: Wiener Filter Coefficients**

Grouping: Grouping in the second step is in some ways similar to the grouping in the first step; but here the power spectrums of the first step are grouped, not just the patches from the noisy image. The same formula is used:

$$3D\,S_i^{Wiener} = \left\{ j \in \Omega : \frac{\left\| \hat{u}_{S^{hard}_{[v]_i}} - \hat{u}_{S^{hard}_{[v]_j}} \right\|_2^2}{\left(N_1^{Wiener}\right)^2} \leq T^{Wiener}_{match} \right\} \tag{2.47}$$

where $\hat{u}_{S^{hard}_{[v]_i}}$ and $\hat{u}_{S^{hard}_{[v]_i}}$ are the estimated sub-images from Step One at locations i and j, respectively. At this stage there are two groups: (1) a group of similar patches derived from the noisy image and (2) a group of similar patches derived from the first step.

Collaborative Filtering: After grouping the patches, a 3D transformation is applied to the 3D array of the grouped patches. Wiener shrinkage is applied to the transformation coefficients of the 3D array. The definition of the Wiener shrinkage coefficients of the power spectrum of the first step is shown in the Equation:

$$3D\,W_{S_i^{Wiener}} = \frac{\left|T_{3D}^{Wiener}\left(3D\,S_i^{Wiener}\right)\right|^2}{\left|T_{3D}^{Wiener}\left(3D\,S_i^{Wiener}\right)\right|^2 + \sigma^2} \tag{2.48}$$

where T_{3D}^{Wiener} is the 3D linear transformation and $3D\,S_i^{Wiener}$ is the result of Equation 2.47. The final stage in the collaborative Wiener filtering of the second step is to multiply the Wiener shrinkage coefficients element-by-element by the 3D transformation coefficients of the noisy image. Here, the inverse of the 3D transformation is applied. This multiplication and the inverse of the 3D transformation are shown in the Equation:

$$3D\,\hat{u}_{S_i^{Wiener}} = T_{3D}^{Wiener^{-1}}\left(3D\,W_{S_i^{Wiener}} \times \left(T_{3D}^{Wiener}\left(3D\,v_i\right)\right)\right) \tag{2.49}$$

where $3D\,v_i$ is the 3D transform coefficients of the noisy data.

Aggregation Weights: Adjusting the weights in this step is not like first step, which depends on the number of non-zero coefficients reached after the hard-thresholding. The weights, here, depend on the Wiener shrinkage coefficients; the weights are assigned as:

$$\omega_i^{Wiener} = \sigma^{-2}\left\|W_{S_i^{Wiener}}\right\|_2^{-2}. \tag{2.50}$$

The parameters set for the original BM3D are shown in Table 2.1 on the next page, which has the following parameters:

N_{step1}^{hard}: **step size for searching the patches inside the search window,**

N_{step2}^{hard}: **step size for moving the search window,**

$N_{Prev.}^{hard}$: **small search window width for fast BM3D,**

β^{hard}: **the Kaiser window function one parameter for reducing the borders effect,**

Table 2.1: Parameter set for the original BM3D filter

Step	Symbol	Description	Fast BM3D	Normal BM3D	
				$\sigma \leq 40$	$\sigma \geq 40$
Step 1 (hard) parameters	T^{2D}_{hard}	2D transformation	2D-Bior1.5	2D-Bior1.5	2D-DCT
	T^{3D}_{hard}	3D transformation	1D-Haar	1D-Haar	1D-Haar
	N^{hard}_{1}	Patch size	8	8	12
	N^{hard}_{2}	3D array size	16	16	16
	N^{hard}_{step1}	Patch step size	6	3	4
	N^{hard}_{search}	Window size	25	39	39
	N^{hard}_{step2}	Window step size	6	1	1
	$N^{hard}_{Prev.}$	Small searching window	3	-	-
	β^{hard}	Kaiser window parameter	2.0	2.0	2.0
	λ_{2D}	2D thresholding operator	0	0	2.0
	λ_{3D}	3D thresholding operator	2.7	2.7	2.8
	T^{hard}_{match}	Similarity distance	2500	2500	5000
Step 2 (Wiener) parameters	T^{2D}_{Wiener}	2D transformation	2D-DCT	2D-DCT	2D-DCT
	T^{3D}_{Wiener}	3D transformation	1D-Haar	1D-Haar	1D-Haar
	N^{Wiener}_{1}	Patch size	8	8	11
	N^{Wiener}_{2}	3D array size	16	32	32
	N^{Wiener}_{step1}	Patch step size	5	3	6
	N^{Wiener}_{search}	Window size	25	39	39
	N^{Wiener}_{step2}	Window step size	5	1	1
	$N^{Wiener}_{Prev.}$	Small searching window	2	-	-
	β^{Wiener}	Kaiser window parameter	2.0	2.0	2.0
	T^{Wiener}_{match}	Similarity distance	400	400	3500

Haar: **a Haar transformation, and**

Bior1.5: **a biorthogonal wavelet.**

- **Recent Works in BM3D**

Inspired by the success of BM3D, many image denoising researchers have proposed a number of modifications that are designed to improve the performance of BM3D. The BM3D filters has two filtering steps and more than twenty parameters. Improving the way of adjusting any of the twenty parameters would improve the output of the BM3D method. Some of the modifications addressed the collection of similar patches; some of them used the transformation of the 3D grouped patches, and others focused on the aggregation of the patches before each stage of the BM3D's two stages. Suwabe *et al.* [93] modified the way of collecting similar patches in the BM3D from non-locally to globally. They proposed using iterative filtering with *Chebyshev polynomial approximation*

Table 2.2: All of the BM3D filter modifications. Four main categories based on the modification application for illustrative purposes.

BM3D Modifications				
Year	Grayscale Images	Colour Images	Video	Applications
	Name	Name	Name	Name
2007	BM3D-SH2D	C-BM3D	V-BM3D	-
2008	BM3DUPSA1	RI-BM3D	-	-
	BM3DUPSA2	-	-	-
2009	BM3D-SPCA	CC-BM3D	CV-BM3D	-
	BM3D-MAD	-	-	-
2010	AL-BM3D	-	-	-
	BM3D-SSBS	-	-	-
2011	BM3D-AD	CF-BM3D	BM3D-T	R-BM3D
	BM3D-Hou	-	-	PT-BM3D
	BM3D-frames	-	-	-
2012	IDD-BM3D	-	V-BM4D	OC-BM3D
	BM3D-DL	-	-	CT-BM3D
2013	-	-	-	BM4D
2014	Fast-BM3D	-	-	-

(CPA) in order to collect the patches from the whole noisy image. Bashar *et al.* [12] replaced the fixed hard thresholding scheme by applying a learning-based adaptive hard thresholding scheme that takes into consideration the context of corresponding blocks. Hasan *et al.* [47] improved the Wiener filter of BM3D by maximizing the *structural similarity* between the patches instead of using the *mean square error*. Moreover, they introduced a 3D zigzag thresholding. However, since 2007, BM3D has been citied in 329 publications based on IEEE; 27 publications out of the 329 are actually considered to be modifications of the original BM3D. For simplicity, the 27 modifications are categorized based on their applications into four categories and appear in Table 2.2 as follows: (1) grayscale image denoising; (2) colour image denoising; (3) video denoising, and (4) other applications such as medical image denoising. According to Table 2.2, about half of the modifications were proposed for the denoising of grayscale images. When denoising colour images, the authors considered the similarity between the levels of the colour in red, green, and blue (RGB). For denoising a video, the similarity between the frames is addressed. In this section, the BM3D for grayscale image modifications are discussed.

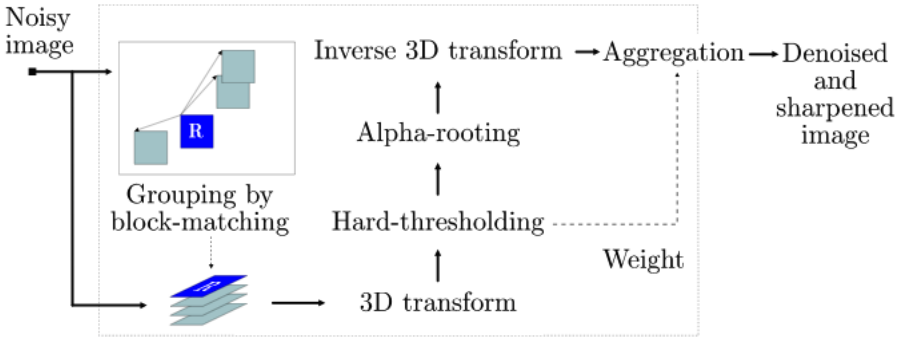
BM3D-Hou: The BM3D performance decreases sharply when the standard deviations of the noise reaches 40. In the BM3D-Hou approach, which is a parameters modification approach, it is argued that adjusting some of the BM3D parameters would help in resolving this level of performance

drop [51]. The maximum grouped patches in BM3D are restricted to $N_2^{hard} = 16$ by default. In the BM3D-Hou approach it is suggested that N_2^{hard} should be greater than 16, such as $N_2^{hard} = 32$, when the noise level is high. Hence, the maximum distance between two similar patches T_{match}^{hard} for the thresholds should be large enough to ensure that there are enough patches in the grouped patches. In the BM3D-Hou approach it is suggested that $T_{match}^{hard} = 25000$ instead of $T_{match}^{hard} = 5000$ in the original BM3D. Indeed, [64] ran an experiment on the influence of the thresholds of T_{match}^{hard} and the results showed that BM3D performs better when $T_{match}^{hard} = 25000$ and the standard deviation of the noise reaches 90 and 100. The BM3D-Hou approach decreases the size of the patches from $N_1^{hard} = 12$ to $N_1^{hard} = 8$. Where BM3D-Hou used $\lambda_{2D} = 0$ instead of $\lambda_{2D} = 2.0$, λ_{2D} defines the thresholding level of the 2D patches in the transform domain during the first filtering stage. The experimental results of BM3D-Hou show more encouraging results than the BM3D when the noise standard deviation reaches 40.

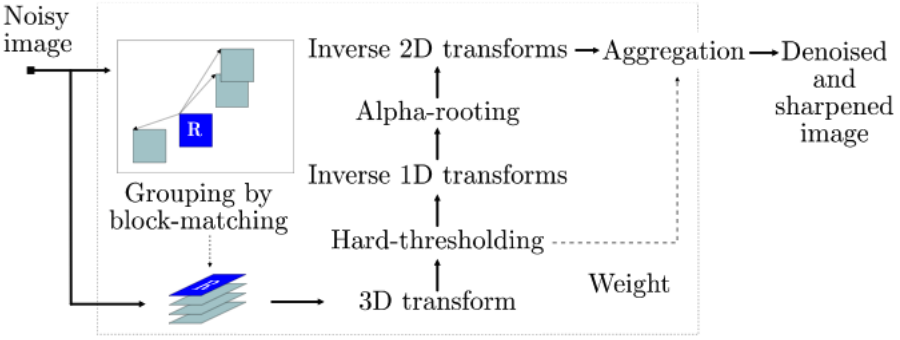
BM3D-SH2D and BM3D-SH3D: Many traditional image sharpening techniques exist. Those techniques rely on the filters to boost the image to higher frequencies such as in the histogram-based methods. Other techniques exploit frequency transformations to improve their efficiency, where the sharpening is accomplished by exaggerating parts of the transformation spectrum. One of these techniques is the alpha-rooting [4] technique, where the α-root of the original coefficient's magnitude is taken for a constant that is larger than 1. One of the drawbacks to the sharpening approach is the exaggeration of the noise components, which exists in some of the images while sharpening. In order to overcome this problem, *BM3D sharpening 2D* (BM3D-SH2D) and *BM3D Sharpening 3D* (BM3D-SH3D) were proposed as a BM3D modification by [25]. This modification permits the sharpening of edges and reveals the fine details in images while eliminating noise components. This modification computes the alpha-rooting for the transformed spectrum of the signal as:

$$T_{SH} = \begin{cases} sign\left[t\left(i\right)\right]\left|t\left(0\right)\right|\left|\frac{t(i)}{t(0)}\right|^{\frac{1}{\alpha}}, & \text{if} \rightarrow t\left(0\right) \neq 0 \\ t\left(i\right), & otherwise \end{cases} \tag{2.51}$$

where $t\left(0\right)$ is the transformed coefficient of the signal; $t\left(\mathrm{i}\right)$ is the original signal and $\alpha > 1$ is an exponent of the control's amount of sharpening. While BM3D only performs thresholding in the first stage, both BM3D-SH2D and BM3D-SH3D apply (Equation 2.51) to the results of thresholding in the first stage. Utilizing alpha-rooting helps in resolving the problem of exaggerating the noise components when thresholding. Figure 2.21 shows where the alpha-rooting is applied in BM3D-SH2D and BM3D-SH3D. Here, BM3D-SH3D applies alpha-rooting after the 3D transformed spectra, BM3D-SH2D inverses the 1D transform before the application of the



(a)



(b)

Figure 2.21: BM3D-SH2D and BM3D-SH3D flowcharts: (a) BM3D-SH2D with alpha-rooting performed on 2D spectra, and (b) BM3D-SH3D with alpha-rooting performed on 3D spectra.

alpha-rooting.

Using weights that are inversely proportional to the total variance aggregates the overlapped patches in BM3D. Unlike BM3D, BM3D-SH3D uses a different method to estimate the variance as:

$$\begin{aligned} var\{T_{SH}\} &\simeq \left(1-\frac{1}{\alpha}\right)^2 |t(0)|^{-\frac{2}{\alpha}} |t(i)|^{\frac{2}{\alpha}} \sigma^2 + \frac{1}{\alpha^2} |t(i)|^{\frac{2}{\alpha}-2} |t(0)|^{\frac{2}{\alpha}-2} \sigma^2 \\ &\simeq \omega_i \sigma^2 \end{aligned} \tag{2.52}$$

and the total variance of the 3D group is estimated as:

$$\upsilon = \sigma^2 + \sum_{t(i)\neq 0, i>0} \omega_i \sigma^2 \tag{2.53}$$

BM3D-UPSA1: An iterative upsampling algorithm called (BM3D-UPSA1) is proposed to improve on BM3D by Danielyan *et al.* [28]. The proposed method produces denoised images with sharp

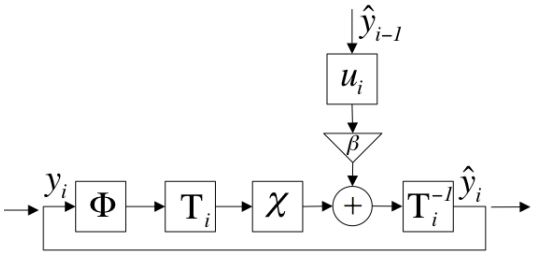


Figure 2.22: The iterative up-sampling scheme for BM3D-UPSA1

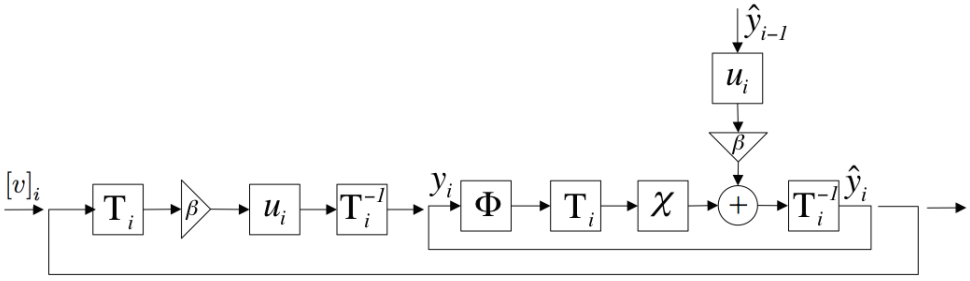


Figure 2.23: The iterative up-sampling scheme for BM3D-UPSA2

edges and fewer artifacts. Here BM3D-UPSA1 replaces the Winner filter stage in BM3D by applying an iterative upsampling algorithm; BM3D-UPSA1 works by upsampling the low-resolution image obtained via the thresholding (BM3D first stage). Figure 2.22 shows the BM3D-UPSA1 scheme. In Figure 2.22, Φ is the result of BM3D first stage is shown, T_i is an orthogonal transform, χ is the shrinkage function, u_i is the result of the previous iteration and β is a zero padding operator. The orthogonal transform consists of a 2D DCT and 1D Haar transform. The iteration process stops after 30 iterations. A filter size of 8×8 is used for the first 7 iterations; then the filter size is reduced to 5×5 . The searching window size used is 25×25.

BM3D-UPSA2: The BM3D-UPSA2 method of Danielyan *et al.* [27] is a modification of BM3D-UPSA1. This method employs the concept of an NL-Means where a non-local patch is gathered in order to enhance the approximation, to be subsequently embedded in the iteration scheme shown in Figure 2.22. The flowchart in Figure 2.23 shows how non-local patches are considered iteratively. The $[\tilde{u}]_i^{hard}$ is the neighbour patches from step one. Twenty iterations with a filter size of 8×8 are used in this method, 25×25 is the searching window size, 2D DCT transformation and 1D Haar transformation are used as orthogonal transformations. BM3D-UPSA2 is used for a filtering video as well. For video denoising, frames are used instead of patches.

BM3D-SSBS: *Block matching* (BM) in BM3D becomes unreliable when the noise level is high, where the transformation coefficients will be less sparse than expected. Therefore, the thresholding

process enforces fine smooth image details and features. The *smooth sigmoid shrinkage function* (SSBS) for the BM3D method (BM3D-SSBS) makes the thresholding in BM3D more robust when BM errors occur due to the high level of noise [86].

The main idea behind thresholding is to split the transformation of the coefficients into signal and noise. The original BM3D uses hard thresholding to set the low amplitudes to zero. A soft thresholding not only sets the low amplitudes to zero but also it shrinks the high amplitudes by the threshold amount, which could be used as a shrinkage function. Because hard thresholding produces some unpleasant discontinuity, soft thresholding is more preferred. The drawback of the soft thresholding is the amount of shrinkage of the high amplitudes that could be signal rather than noise. The BM3D-SSBS function is intended to use smooth sigmoid shrinkage [10] for BM3D in order to retain the advantages of the soft/hard thresholding-flexible shrinkages. The SSBS function is defined as:

$$\delta_{t,\tau,\lambda}(x) = \frac{sgn(x)(x-t)}{1+e^{-\tau(x|\lambda)}} \tag{2.54}$$

where the $\delta_{t,\tau,\lambda}(x)$ function is the product of the soft thresholding when used in conjunction with the sigmoid function; this is why it is called a "smooth" sigmoid shrinkage function. The $\delta_{t,\tau,\lambda}(x)$ function has three degrees of freedom "parameters": t, λ and τ. Here t, which is called the assumption attenuation parameter, controls the amount of shrinkage imposed by the large amplitudes; λ, is called the thresholding height, and serves as a threshold; τ has a geometric interpretation according to the t and λ chosen values. Here the τ parameterizes the curvature of the arc of the SSBS function in the interval (t,λ). In other words, τ controls the amount of shrinkage applied to those coefficients whose amplitudes fit within the interval (t,λ). When λ's value is less than t, the function $\delta_{t,\tau,\lambda}(x)$ performs as a soft thresholding function. As the λ's value is very large, $\delta_{t,\tau,\lambda}(x)$ works as a hard thresholding function where λ is the height of the threshold.

When the signal is not sparse enough due to the high level of noise, BM3D-SSBS replaces the hard thresholding using the SSBS function. A simple sparsity measure is used: 1) first, the number of the coefficients that is greater than a fixed threshold is computed; 2) if the number is large, SSBS is used, otherwise, hard thresholding is used.

BM3D-SAPCA: As a modification to the BM3D, BM3D with *shape-adaptive principal component analysis* (BM3D-SAPCA) was proposed in order to increase the data sparsity [26]. Unlike the original BM3D, BM3D-SAPCA has only one stage for filtering. Shape-adaptive neighbourhoods

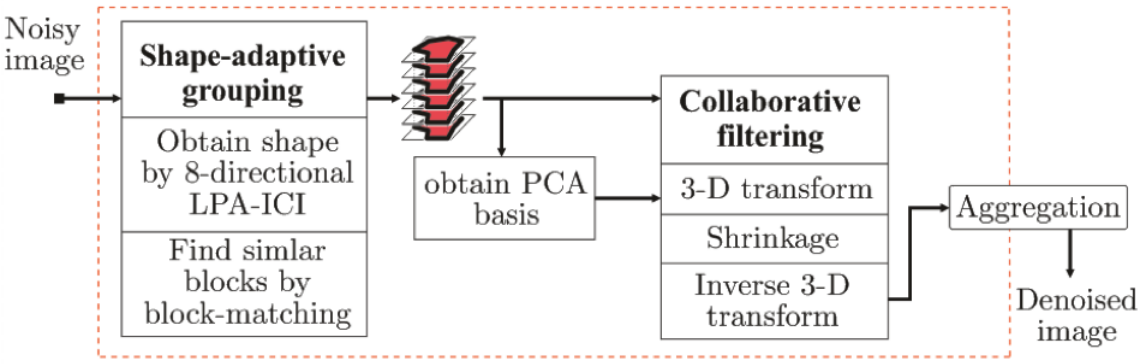


Figure 2.24: BM3D with *shape-adaptive principal component analysis* scheme

are used to ensure that more correlated sub-images are gathered instead of using blocks as in BM3D. Here BM3D-SAPCA applies a fixed search window as the BM3D.

Motived by the success of the *local polynomial approximation* achieved by using the *intersection of the confidence interval* rule (LPA-ICI) image denoising [40], BM3D-SAPCA obtains the adaptive-shape neighbourhoods by using 8-directional LPA-ICI. After grouping the adaptive-shape neighbourhoods, a fixed threshold is used to determine whether to apply the PCA or to apply the collaborative hard thresholding as in the BM3D.

Here BM3D-SAPCA uses the PCA as a part of the 3D transformation. A threshold is used for the PCA eigenvalues in order to select eigenvectors. The threshold is proportional to the noise variance. However, utilizing PCA for image denoising has been extensively discussed in Section 2.2.2.4 on page 35. Figure 2.24 shows the BM3D-SAPCA scheme.

2.3 Multi-view Images Denoising

The *human visual system* (HVS) observes objects through a binocular (two eyes) system, and then it directly maps different views (images) of the objects into our brain. The views are different because each eye is observing a slightly different image from a different angle. By comparing the relative size (binocular disparity) between the two image objects, the depth of the information can be extracted. Figure 2.25 shows a scene as seen from different eyes (angles).

At present, there are several approaches to extracting depth information by computing the deference between two images (stereoscopic images). One popular approach can be seen at the cinema where polarized lights are used to project images from two different angles onto one screen. The

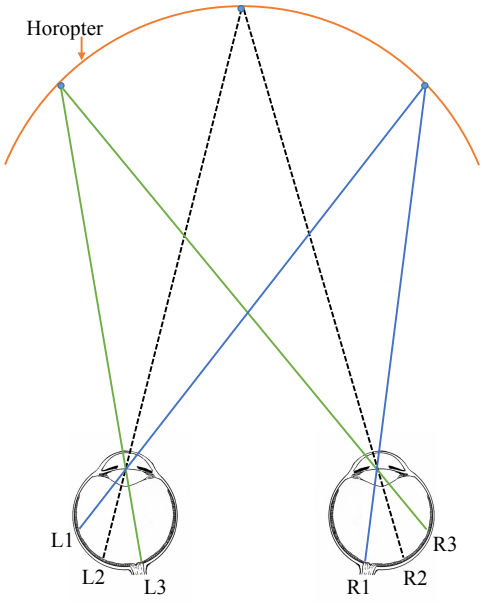
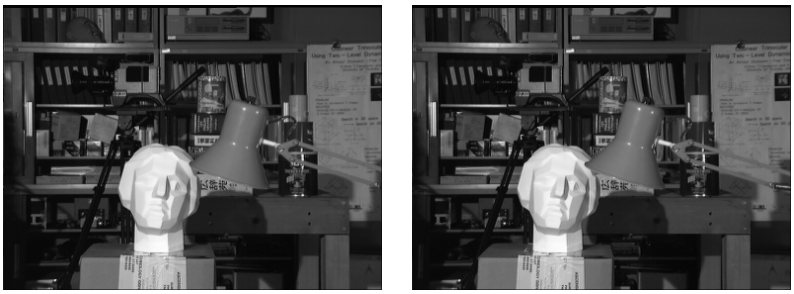


Figure 2.25: The horopter points are imaged at corresponding areas in the two retinas.



Left Right

Figure 2.26: An example of a stereoscopic image

viewers use glasses where each lens accepts one of the polarized lights, and block the other. One scene is watched but two different images are received.

Stereoscopic images can also be exploited for image segmentation. The depth of information extracted from a stereoscopic image can be utilized further for segmenting objects. Boykov *et al.* [16] used a graph-cut algorithm for segmenting objects from stereoscopic images. An example of a stereoscopic image is shown in Figure 2.26.

2.3.1 Multiple-view Image Denoising Using PCA

Zhang *et al.* showed that using multi-view images for image denoising has many benefits over using a single-view image [107]. A noisy pixel in a multi-view image is estimated based on the corresponding pixels from all of the other images. In addition, they extended the idea of using patch-based PCA denoising from a single image to multi-view image denoising. In the method, similar patches are collected locally and globally from the multiple images before applying the PCA algorithm for filtering.

2.3.2 Maximum a Posteriori-Markov Random Field

Maximum a posteriori-markov random field (MAP-MRF) is utilized by Heo *et al.* [49] as a model for energy minimization in order to compute the disparity maps from a multi-view image. They proposed an algorithm that initially restored intensity differences by adapting an NL-Means algorithm. Then, the dissimilarity of support pixel distributions was calculated, where the *mean square error* was utilized to group similar patches.

2.3.3 A Statistical Approach with Adaptive NL-Means

Luo *et al.* [69] utilized an adaptive NL-Means algorithm with a joint-view distance for multi-view image denoising. They replaced the fixed number of similar patches with an adaptive one based on the variance of the collected patches. They exploited the off-the-shelf algorithms of Chan *et al.* [19] as a joint-view distance to compute the correspondences (disparity maps).

2.3.4 Discussion

In this section, a short background for the use of multi-view images in denoising is presented. The multi-view image denoising methods can be improved through applying the approach used for computing correspondences between image views, and the denoising algorithm. Some of the methods utilize simple image similarity metrics, and others exploit statistically based methods for computing image similarity. Some methods adapt NL-Means and others employ PCA as denoising algorithms. Multi-view images improve the performance of the denoising by exploiting the redundancy of the different views.

2.4 Additive White Gaussian Noise Estimating

Estimating subjectively the true noise level of an image in real-life situations is a challenge. Therefore, embedding noise level estimating techniques into image denoising methods is highly needed. Various techniques of noise estimation were proposed. Noise estimating techniques are grouped into two main categories: spatial domain techniques, and frequency domain techniques. In this following subsections, we present the results of evaluating these techniques.

2.4.1 Spatial Domain Estimation Techniques

2.4.1.1 Local Mean and Variance

Estimation based on the local mean and variance [57, 44]: in this technique, we estimated the local mean and variance in uniform patches as extracted from the entire image as:

$$u(x,y) = \frac{1}{n \times n} \sum_{x,y \in P} I(x,y) \tag{2.55}$$

and

$$\sigma^2(x,y) = \frac{1}{n \times n} \sum_{x,y \in P} I(x,y)^2 - u(x,y)^2 \tag{2.56}$$

where P is the local squared patch with size $n \times n$, $u(x,y)$ is local mean, and $\sigma^2(x,y)$ is the local variance. All local variance estimates are then averaged. The noise standard deviation equals to $\sigma = \frac{1}{M \times N} \sqrt{\sum \sigma^2(x,y)}$. $M \times N$ is the input image size. The local mean and variance estimation technique utilizes locally squared kernel. We tested this technique using various kernel sizes (2×2, 4×4, 12×12, and 32×32), we have found that this technique produces closest estimate to the true noise level when we applied 12×12 kernel size.

2.4.1.2 Sum of the Absolute Values of the Laplacian

Estimation based on sum of the absolute values of the Laplacian [53]: in this estimator, we applied laplacian mask first, then we computed sum of the absolute values of Laplacian as

$$N = \begin{array}{|c|c|c|}\hline 1 & -2 & 1 \\ \hline -2 & 4 & -2 \\ \hline 1 & -2 & 1 \\ \hline \end{array}$$

and the standard deviation is computed as

$$\sigma = \sqrt{\frac{\pi}{2}} \frac{1}{6\left(M-2\right)\left(N-2\right)} \sum_{I} \left| I\left(x,y\right) \circledast N \right|. \tag{2.57}$$

where $\circledast$ is the convolution.

2.4.1.3 Simplified Noise-power Spectra Estimating

Simplified method of estimating noise-power spectra [46]: we applied the mask N first in this estimator, then we computed its convolution with the noisy image as

$$N = \begin{array}{|c|c|c|}\hline 0 & -0.2500 & 0 \\ \hline -0.2500 & 1 & -0.2500 \\ \hline 0 & -0.2500 & 0 \\ \hline \end{array}$$

and

$$eps = \sqrt{\frac{4}{5}} \times \left| I\left(x,y\right) \circledast N \right|, \tag{2.58}$$

the standard deviation is computed as

$$\sigma = \sqrt{\frac{1}{M \times N} \times \sum_{I} eps^2}. \tag{2.59}$$

2.4.2 Frequency Domain Estimation Techniques

2.4.2.1 Wavelet Median Absolute Deviation

The median absolute deviation of the wavelet coefficients [35, 34]: by using this technique, we estimated the noise variance by computing the median absolute deviation of the wavelet coefficients at the finer scale as:

$$\sigma = median_{\eta} \left| T^{2D} I\left(x, y\right)_i - median_i \left(T^{2D} I\left(x, y\right)_i \right) \right| \times \beta \tag{2.60}$$

where $T^{2D} I\left(x, y\right)_i$ is the 2D linear transformation of a pixel located at row i from the noisy image, $median_i$ the median of values located at row i, $median_{\eta}$ is the median of medians of all rows, and β is a rescaling parameters equals to 1.4826.

2.4.2.2 PCA for Noise Estimating

Noise level estimation based on PCA [110, 66, 67]: in this approach, the standard deviation of the additive noise is estimated based on the covariance of weak textures patches in the frequency domain. PCA transformation is deployed as a transform. The covariance of the patches, is calculated iteratively by utilizing a gradient decent optimization method for finding a local minimum variance. At each iteration n, the following steps from 1 to 4 are processed:

1. The high textures patches are excluded using a threshold at each iteration. The inverse gamma cumulative distribution function tunes the level of thresholding as:

$$\tau = \sigma_n^2 F^{-1}\left(\delta, \alpha, \beta\right) \tag{2.61}$$

   where σ_n^2, which is the variance, is the eigenvector associated to the minimum eigenvalue of the covariance matrix of the week textures patches of the noisy image at the iteration n, δ is level of confidence equals to 0.999, $\alpha = \frac{patch\ size^2}{2}$, and $\beta = \frac{patch\ size^2}{2} \times tr\left(D_h^{'} \times D_h + D_v^{'} \times D_v\right)$ where tr is the trace of D_h and D_u matrices represent horizontal and vertical derivative operators, respectively.

2. Computing the covariance matrix of the weak texture patches.

3. Obtaining the variance σ_n^2, which is equal to the minimum eigenvalue of the covariance

matrix of the weak texture patches. The initial σ^2 is set to the minimum eigenvalue of the covariance matrix of the entire noisy patches.

4. Finally, the standard deviation σ of the additive noise equals to the square root of the variance when reaching the local minimum variance.

2.4.3 Discussion

We contaminated our grey-scale dataset with AWGN, and utilized the noise estimating techniques presented to estimate the noise level σ . The noise level ranges between 10 to 100 were found. We used the best parameters that we found for each method, for instance, 5×5 patch size was employed for local mean and variance technique. After running the experiment, we have found that frequency domains filters outperform the spatial domain filters. Noise level estimation based on PCA technique of Xinhao *et al.* [67] have achieved the closest estimate. The chart shown in Figure 2.27 illustrates the results of the experiment when estimating noise in the `Boat` image.

2.5 Images Similarity Measures

Evaluating qualitatively, or subjectively, the performance of the various image denoising methods would vary from person to person. Therefore, it is essential to use quantitative image similarity measures along with the subjective evaluation in order to compare the quality of the outcomes of these denoising methods. Image similarity measures produce quantitative values that represent the degree of matching among images, or patches.

Image similarity measures may vary. Some of them are very sensitive to additive noise, others are more sensitive to speckle noise, and still others are only sensitive to luminance information. Also computing image similarity could be either global or local. In global computing, a measure returns a value that describes the similarity of the whole image. Local similarity measures, on the other hand, divide the images into blocks before computing the similarity for each block separately.

Akramullah [6] stated that "Human eyes are the most sensitive to luma rather than chroma information in digitally coloured images". For this reason, it is preferred to apply image similarity measures in the luma channel via a colour scheme that separates the intensity of the luma channel from the chroma channel; for instance, the Y' channel in the Y'CbCr colour space. However, the used images in this thesis is a grey-sale images.

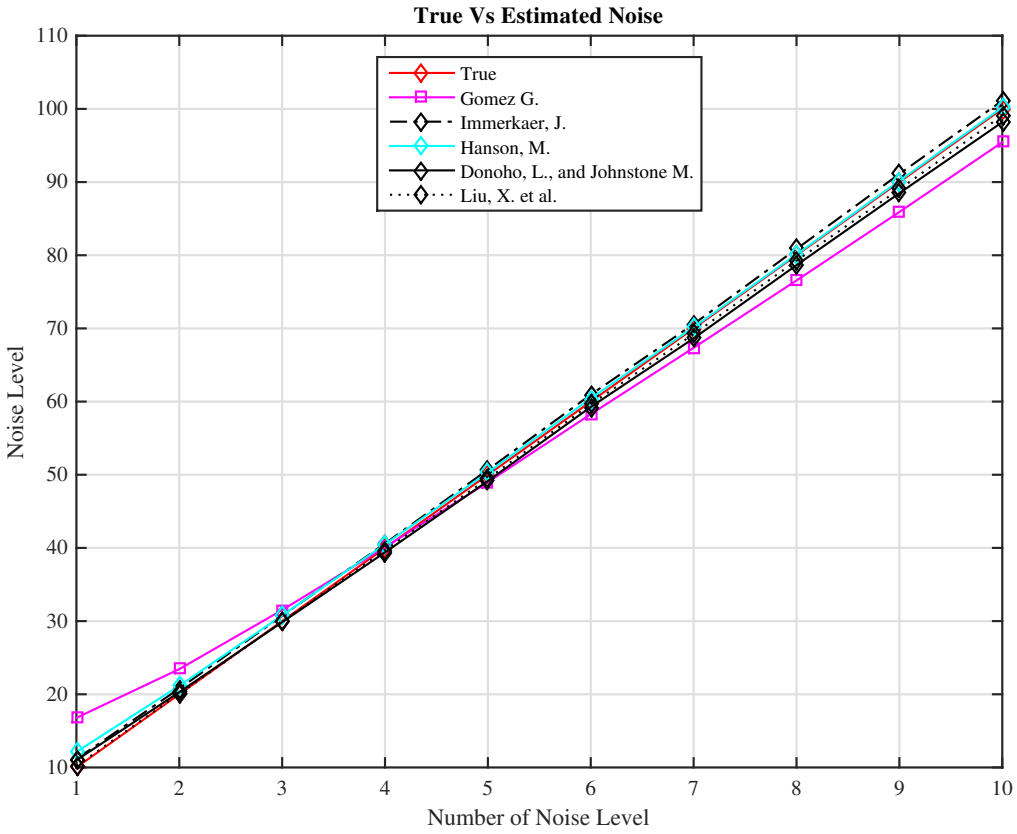


Figure 2.27: Estimating AWGN in the noisy `Boat` image: Gomez G. [44], Immerkaer, J. [53], Hanson, M. [46], Donoho, L., and Johnstone M. [35], Liu *et al.* [66].

If we assume that $u(x, y)$ is a true image of size $M \times N$ with the coordinates (x, y), and that $\hat{u}(x, y)$ is the filtered image, a similarity measure could be mathematically represented as

$$S\left(u(x, y), \hat{u}(x, y)\right). \tag{2.62}$$

Most of similarity measures $S\left(u(x, y), \hat{u}(x, y)\right)$ are image similarity metrics when they satisfy the following constraints:

1. Boundedness $S\left(u(x, y), \hat{u}(x, y)\right) \geq 0$
2. Symmetry $S\left(u(x, y), \hat{u}(x, y)\right) = S\left(\hat{u}(x, y), u(x, y)\right)$
3. [1]Uniques minimum $S\left(u(x, y), \hat{u}(x, y)\right) = 0,$ only if $\rightarrow u(x, y) = \hat{u}(x, y)$

This section provides insight into some of the image similarity measures.

2.5.1 Mean Square Error

Mean square error (MSE) is one of the simplest and most common similarity measures. The MSE measure represents the average of the squares of the difference "error" between the referenced and the filtered images. The higher the value of MSE, the poorer the quality of the denoised image. The MSE measure is sensitive to the outliers' pixels. The MSE is given as:

$$MSE = \frac{1}{M \times N} \sum_{x=1}^{M} \sum_{y=1}^{N} \left(u\left(x, y\right) - \hat{u}\left(x, y\right)\right)^2 \tag{2.63}$$

2.5.2 Peak Signal to Noise Ratio

PSNR is an expression that represents the ratio between the maximum possible value of a signal and the value of the distorting noise that affects the quality of its representation. The higher the PSNR's value, the better the quality of the denoised image. According to Mitchell [76], because MSE and PSNR are sensitive to outliers' pixels, the search window size for the local measures should be at least 20×20 in order to increase the number of similar patches. The PSNR is defined as:

[1]Some measures use 1 or 100 instead of 0 as an unique maximum when $u(x, y) = \hat{u}(x, y)$

$$PSNR = 10\log_{10}\left(\frac{(2^n-1)^2}{MSE}\right) = 10\log_{10}\frac{255^2}{MSE} \tag{2.64}$$

where n is an integer number representing the number of bits per pixel, $n = 8$ in the case of grey-scale images.

2.5.3 Normalized Cross-Correlation

A *normalized cross-correlation* (NCC) estimates the degree to which two images are correlated. NCC is normalized by the amount of true image energy necessary to give unity as the peak correlation. The NCC measure is sensitive to the outliers' pixels, but it is more robust to changes of illumination than MSE and PSNR. The higher the NCC's value, the higher the quality of the denoised image. The NCC is defined as:

$$NCC = \sum_{x=1}^{M}\sum_{y=1}^{N}\frac{u(x,y)\cdot\hat{u}(x,y)}{u(x,y)^2} \tag{2.65}$$

2.5.4 Mean Average Error

The *mean average error* (MAE) represents the average difference between two images. The higher the value of the MSE, the lower the quality of the denoised image. The MAE is defined as:

$$MAE = \sum_{x=1}^{M}\sum_{y=1}^{N}\frac{|u(x,y)-\hat{u}(x,y)|}{M\times N} \tag{2.66}$$

2.5.5 Structural Content

The *structural content* (SC) estimates the perceived visual quality of denoised images. The higher the value of SC, the higher the quality of the denoised image. The SC is given as:

$$SC = \sum_{x=1}^{M}\sum_{y=1}^{N}\frac{u(x,y)^2}{\hat{u}(x,y)^2} \tag{2.67}$$

2.5.6 Maximum Difference

The *maximum difference* (MD) approximates the maximum difference between the true and the estimated images. The lower the value of MD, the higher the quality of the denoised image. The MD is defined as:

$$MD = \max\left(\left|u\left(x,y\right) - \hat{u}\left(x,y\right)\right|\right) \tag{2.68}$$

2.5.7 Laplacian Mean Square Error

The *laplacian mean square error* (LMSE) is an image similarity measure calculated based on the Laplacian value of the true and estimated images. The lower value of LMSE, the better the quality of the denoised image. LMSE is defined as:

$$LMSE = \sum_{x=1}^{M}\sum_{y=1}^{N}\frac{\triangle\left(u\left(x,y\right)\right) - \triangle\left(\hat{u}\left(x,y\right)\right)}{\left[\triangle\left(u\left(x,y\right)\right)\right]^2} \tag{2.69}$$

where the Laplacian operator is denoted as:

$$\triangle\left(u\left(x,y\right)\right) = u\left(x+1,y\right) + u\left(x-1,y\right) + u\left(x,y+1\right) + u\left(x,y-1\right) - 4u\left(x,y\right) \tag{2.70}$$

2.5.8 Normalized Absolute Error

The *normalized absolute error* (NAE) measures the difference between the true and the estimated images by the normalized absolute error as:

$$NAE = \sum_{x=1}^{M}\sum_{y=1}^{N}\frac{\left|u\left(x,y\right) - \hat{u}\left(x,y\right)\right|}{u\left(x,y\right)} \tag{2.71}$$

The lower the value of NAE, the higher the quality of the denoised image.

2.5.9 Mean Structural Similarity

Given the limitations of the previously discussed objective measures, Wang *et al.* [97] proposed a more extensive image *structural similarity* (SSIM) measure that can evaluate image luminance, contrast, and structural information changes. The luminance component is defined as:

$$l\left(u\left(x,y\right),\hat{u}\left(x,y\right)\right)=\frac{2\mu_{u(x,y)}\mu_{\hat{u}(x,y)}+C_1}{\mu_{u(x,y)}^2+\mu_{\hat{u}(x,y)}^2+C_1}, \tag{2.72}$$

the contrast component is defined as:

$$c\left(u\left(x,y\right),\hat{u}\left(x,y\right)\right)=\frac{2\sigma_{u(x,y)}\sigma_{\hat{u}(x,y)}+C_2}{\sigma_{u(x,y)}^2+\sigma_{\hat{u}(x,y)}^2+C_2}, \tag{2.73}$$

and the structure component is defined as:

$$s\left(u\left(x,y\right),\hat{u}\left(x,y\right)\right)=\frac{\sigma_{u(x,y)\hat{u}(x,y)}+C_3}{\sigma_{u(x,y)}+\sigma_{\hat{u}(x,y)}+C_3}. \tag{2.74}$$

Where $\mu_{u(x,y)}$ and $\mu_{\hat{u}(x,y)}$ are the means of the true reference image patch and the noisy image patch, respectively, and $\sigma_{u(x,y)}$ and $\sigma_{\hat{u}(x,y)}$ are the standard deviations of each image, $\sigma_{u(x,y)\hat{u}(x,y)}$ represents the covariance of the two images, and C_1, C_2, and C_3 are the constants used to avoid instability. The three components are independents. This measure combines the three terms to produce a unique similarity measure:

$$SSIM\left(u,\hat{u}\right)=\left[l\left(u,\hat{u}\right)\right]^{\alpha}\cdot\left[c\left(u,\hat{u}\right)\right]^{b}\cdot\left[s\left(u,\hat{u}\right)\right]^{\gamma}, \tag{2.75}$$

where $\alpha>0$, $\alpha>0$, and $\alpha>0$ are the parameters used to adjust the importance of each component.

Since the luminance and contrast can vary across an image, SSIM locally evaluates the luminance, contrast, and structural changes. It divides the images into patches before assessing each patch by using the combination:

$$SSIM\left(u\left(x,y\right),\hat{u}\left(x,y\right)\right)=\frac{\left(2\mu_{u(x,y)}\mu_{\hat{u}(x,y)}+C_1\right)\left(2\sigma_{u(x,y)\hat{u}(x,y)}+C_2\right)}{\left(\mu_{u(x,y)}^2+\mu_{\hat{u}(x,y)}^2+C_1\right)\left(\sigma_{u(x,y)}^2+\sigma_{\hat{u}(x,y)}^2+C_2\right)} \tag{2.76}$$

The MSSIM evaluates an image's overall quality as:

$$MSSIM\left(u\left(x,y\right),\hat{u}\left(x,y\right)\right)=\frac{1}{M\times N}\sum_{x=1}^{M}\sum_{y=1}^{N}SSIM\left(u\left(x,y\right),\hat{u}\left(x,y\right)\right) \tag{2.77}$$

A study conducted by Horé *et al.* [50] has revealed that MSSIM is less sensitive to additive noise than PSNR. The higher the value of MSSIM, the higher the quality of the denoised image.

2.5.10 Discussion

In this section, various image similarity measures were discussed. The review showed that image similarity measures are application-dependent measures; in other words, there is no universal measure that could be used efficiently for all images processing applications. Image similarity measures vary: some are simple, e.g., MAE or MSE, and others are complicated like MSSIM.

This thesis investigates the quality of the denoising methods outcomes both quantitatively and qualitatively, based on the various sensitivities of the measures. This thesis uses full reference similarity measures. The reference image is assumed to be in existence and free of noise (a true image), as compared with the denoised image. This thesis employs global image similarity measures for comparing the true image to the filtered image. In addition, it considers different local image similarity measures for use in comparing the similarity between patches during the filtering process. Here the required searching window size for local measures must be at least 20×20 when using MSE, or PSNR. The final conclusion in our study is based on the use of MSSIM.

2.6 Discretized Wavelet Transforms

There is a variety of transformations in the transform domain that could be used in the transforming process. One of these transformations, which will be discussed in this section, is the powerful *discretized wavelet transform* (DWT). According to Wei *et al.* [98],"DWT is considered to be a powerful tool for a diversity of digital signal processing applications". The DWT is a simplified version of the *continuous wavelet transform* (CWT).

CWT provides highly redundant information about the transformed signal, but it requires a massive

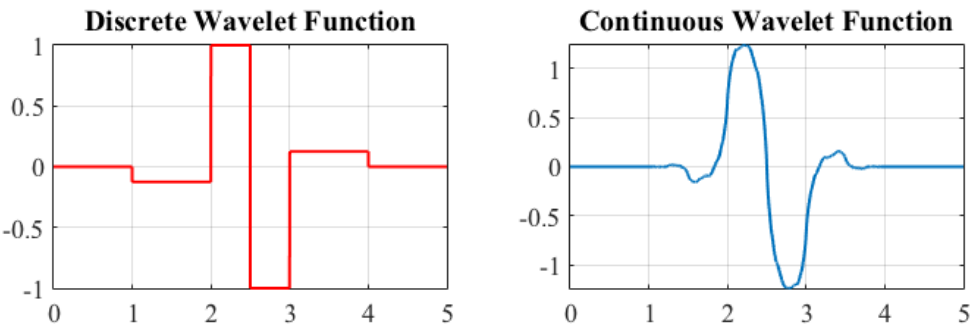


Figure 2.28: Single-level continuous and discrete 1D wavelet transform

amount of resources and computation time. On the other hand, DWT offers acceptable information about the transformed signal, along with a reduction in resources and computation time. Figure 2.28 shows a single-level continuous and discrete 1D wavelet transform.

Qualities of the wavelet families vary according to several criteria. Understanding the criteria is essential for choosing an optimal transform for image application. The criteria include:

1. Support quantifying both time and frequency localizations.
2. The symmetry or anti-symmetry. In image processing, symmetry helps in avoiding dephasing.
3. Number of vanishing moments for both the scaling or wavelet functions. Wavelets with large numbers of vanishing moments produce sparse representation for large signal "images".
4. The regularity of the wavelet. The regularity assists in smoothing the reconstruction image.
5. Orthogonality or bi-orthogonality of the transforms.

Discretized wavelet transform has several families. These families include: Haar, Daubechies, Symlets, Coiflets, Biorthogonal, Reverse Biorthogonal, and Meyer wavelets. The properties information of the wavelet families are explained in Table 2.3.

2.6.1 Haar Wavelet

In 1909, Alfréd Haar [45] presented the first and simplest wavelet basis, which was called a Haar wavelet. A Haar wavelet is a discontinuous step function, and therefore it is not differentiable. The Haar wavelet function is similar to the Daubechies `db1` function. The wavelet function of Haar is shown in Figure 2.29.

Table 2.3: The properties information of the wavelet families. Here * is strictly a positive integer.

Family	Short name	Examples	Order N	Symmetry	Orthogonal
Haar	haar	the same as db1	-	yes	yes
Daubechies	db	db1, ..., db**	N = 1, 2, ...	far from	yes
Symlets	sym	sym2, ..., sym**	N = 2, 3, ...	near from	yes
Coiflets	coif	coif1, ..., coif5	N = 1, ..., 5	near from	yes
Biorthogonal	bior	bior1.1, ..., bior1.5	N = 1, 3, 5	yes	no
		bior2.2, ..., bior2.8	N = 2, 4, ..., 8		
		bior3.1, ..., bior3.9	N = 1, 3, ..., 9		
		bior4.4, bior5.5 bior6.8	N = 4, 5, 8		
RBiorthogonal	rbio	rbio1.1, ..., rbio1.5	N = 1, 3, 5	yes	no
		rbio2.2, ..., rbio2.8	N = 2, 4, ..., 8		
		rbio3.1, ..., rbio3.9	N = 1, 3, ..., 9		
		rbio4.4, rbio5.5 rbio6.8	N = 4, 5, 8		
Meyer	meyr	meyr	-	yes	yes
DMeyer	dmey	dmey	-	yes	yes

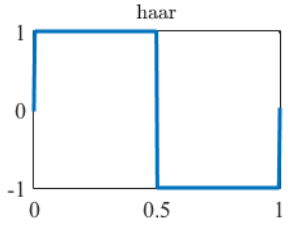


Figure 2.29: The wavelet function Ψ of Haar transform

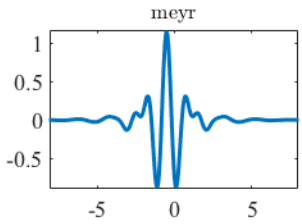


Figure 2.30: The wavelet functions Ψ of Meyer transform

2.6.2 Meyer Wavelet

Unlike the Haar wavelet, the Meyer wavelet [31] is continuously differentiable, but it does not have compact support. The Meyer wavelet is an orthogonal wavelet, which is represented by `meyer`. Figure 2.30 illustrates the wavelet function.

2.6.3 Daubechies Wavelet

Ingrid Daubechies [30] presented the Daubechies wavelets as modifications to the Haar wavelet. The Daubechies wavelets, which are supported by orthonormal wavelets, employ more complicated averages and difference operators than the Haar wavelet. `dbN` presents the name of Daubechies family wavelets, where `N` represents the order of the wavelet. Each member of the family is uniquely identified by its wavelet function. The number of vanishing moments of the wavelet increases when the order of the wavelet `N` increases. `db1` wavelet is similar to the Haar wavelet. The wavelet functions of the Daubechies family wavelets are shown in Figure 2.31.

2.6.4 Symlets Wavelet

The symlets wavelets are nearly symmetrical wavelets proposed by Daubechies [29] as a modification to the Daubechies wavelets but with an increased symmetry. The symlets wavelets' properties are similar to the Daubechies wavelets properties, see the table. `symN` presents the name of symlets wavelets family, where `N` represents the order of the wavelet. Figure 2.32 illustrates the wavelet functions of the symlets' wavelets.

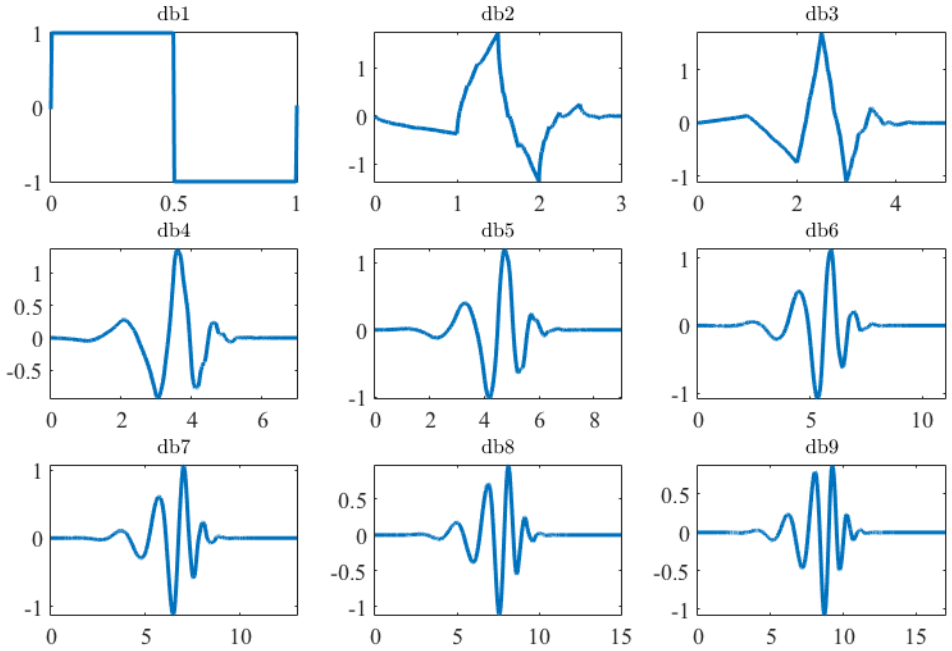


Figure 2.31: The wavelet functions Ψ of Daubechies transform family

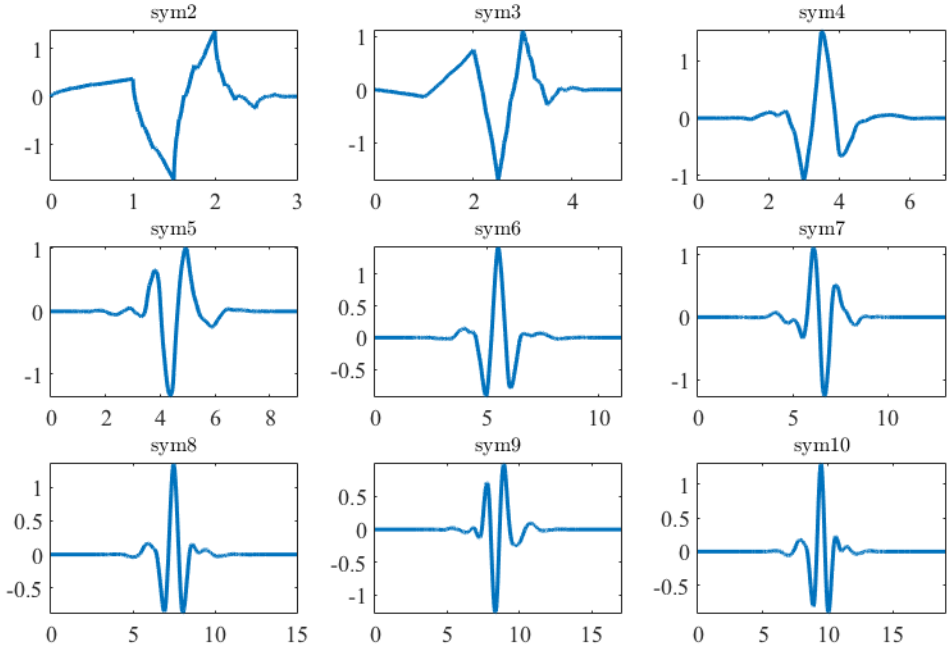


Figure 2.32: The wavelet functions Ψ of Symlets transform family

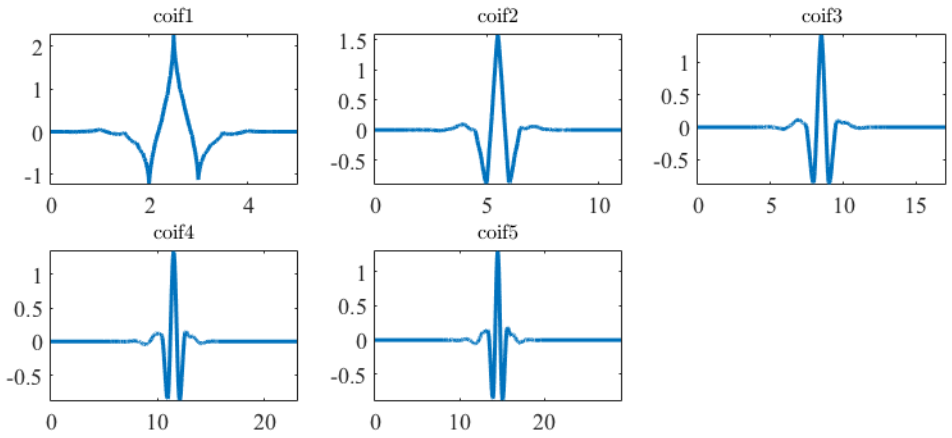


Figure 2.33: The wavelet functions Ψ of Coiflets transform family

2.6.5 Coiflets Wavelet

Coiflets wavelets are near-symmetric wavelets proposed by Ingrid Daubechies [14]. Unlike the symlets wavelets, Coiflets wavelets allow a high number of vanishing moments for both the scaling and the wavelet functions. The Coiflets wavelets are represented by `coifN`, where `N` is the number of vanishing moments for both the wavelet and the scaling functions. The functions of the Coiflets wavelets family are shown in Figure 2.33.

2.6.6 Biorthogonal and Reverse Biorthogonal Wavelets

Biorthogonal [73] and reverse biorthogonal [92] wavelets are symmetric wavelets. Unlike the previously mentioned wavelets, this transform is not an orthogonal wavelet. Biorthogonal and reverse biorthogonal wavelets have two different functions for decomposition and reconstruction; thus, they are not orthogonal except in the single base. Biorthogonal and reverse biorthogonal wavelets are represented by `biorNr.Nd` and `rbioNr.Nd`, respectively; where `Nr` is the number of zeros at pi in the synthesis low pass filter, and `Nd` is number of zeros at pi in the analysis of a low pass filter. The functions of biorthogonal wavelets family are shown in 2.34, and the functions of the reverse biorthogonal wavelets family are shown in Figure 2.35.

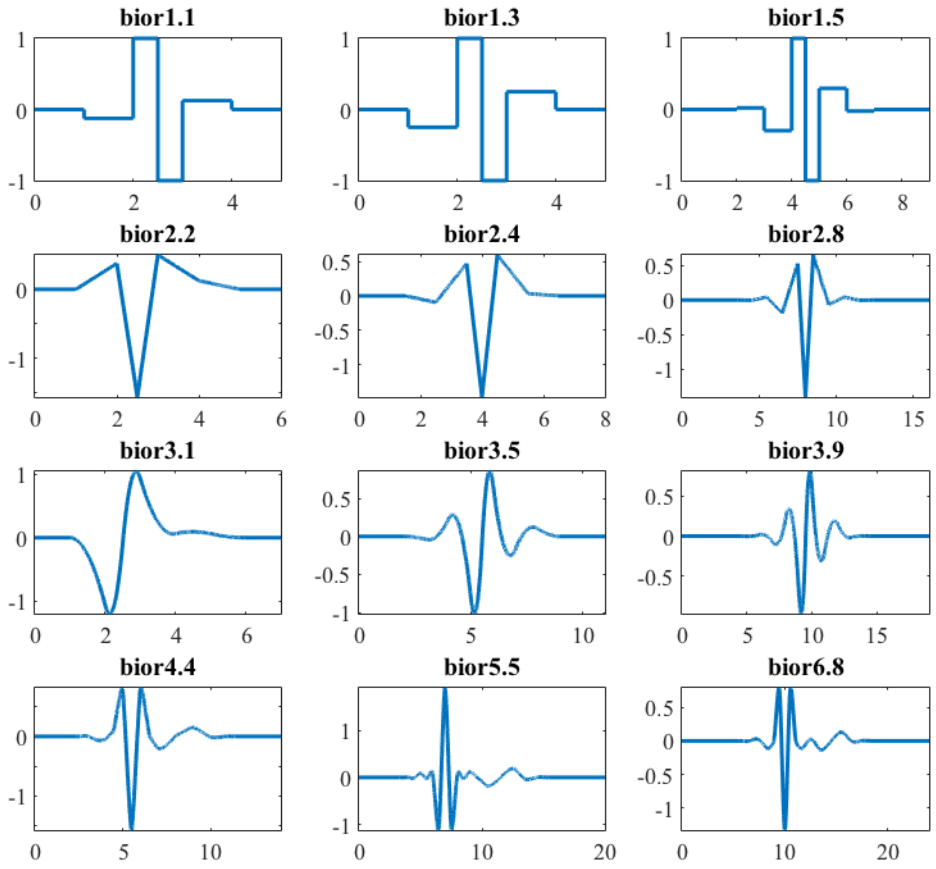


Figure 2.34: The wavelet functions Ψ of Biorthogonal transform family

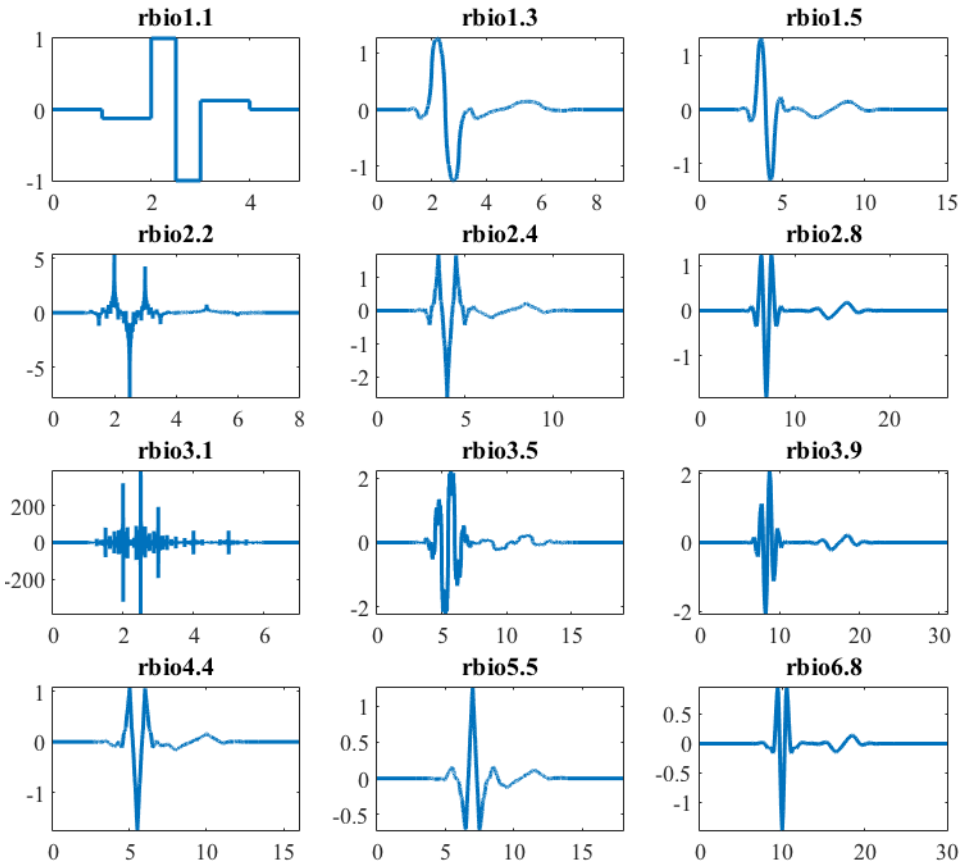


Figure 2.35: The wavelet functions Ψ of Reverse Biorthogonal transform family

2.7 Review Summary

Images are often contaminated with noise during acquisition or transmission or are due to poor quality image compression. The level of noise may vary from being imperceptible to being very noticeable. Image denoising techniques produce a new image that is closer to the original noise-free image. Image denoising methods are categorized into two main groups: single and multi-view image denoising methods. Single image denoising methods process one image, but multi-view image denoising methods process multiple images.

The noise within images can be additive, multiplicative, or a mixture. Since this research presents additive denoising methods, additive denoising methods are explained in greater detail. Additive denoising techniques are grouped into two main approaches: pixel-based image filtering and patch-based image filtering. A pixel-based image filtering scheme manipulates one pixel at a time (pixel-wise) based on its spatially neighbouring pixels. Patch-based image filtering manipulates patches in order to provide an estimate of the true pixel values based on similar patches located within a particular search window. Pixel-based image filtering includes low-pass filters, e.g., the Yaroslavsky filter and BF. Patch-based filters include: NL-Means, the PPB, PB-PCA, DL, and BM3D filter.

Various image similarity measures were previously discussed. There is no universal measure that could be used efficiently in all image processing applications. Some image similarity measures are simple, and others are complicated. Full reference image similarity measures will be used in this work.

Chapter 3

Images Denoising: Empirical Results and Discussion

Unlike the mean filter (see the low-pass filters in Subsection 2.2.1.1), the weighted average of the Gaussian filter assists in gently smoothing noisy images. The weight of this filter is based on the neighbouring pixel's locations. Unfortunately, this kind of filter is not suitable for denoising the edges and the textures due to their high discontinuity nature. The Yaroslavsky filter can only average pixels with close grey levels. It fails to eliminate high noise levels. The mean, Gaussian, and Yaroslavsky filters "low-pass filters" have the capability to denoise AWGN, but they fail to preserve the edges and the fine details. Because the edges have high frequencies, all of the high frequencies are suppressed by the low-pass filters. Several modifications have been proposed to overcome this problem through controlling the amount of smoothing when denoising. BF (in Subsection 2.2.1.2), which is an edge-preserving denoising method, is one of the modifications. Figure 3.1 shows the effect of applying a BF to preserve the edges, while basic low-pass filters do not.

Since the heat equation diffuses isotropically, the heat filter (see the variational denoising filters in Subsection 2.2.1.3) cannot distinguish between noisy pixels and features (i.e., edges). In order to prevent smoothing across edges, a TV filter uses a condition to set the heat equation $\frac{\partial v}{\partial t} = 0$ at near boundaries. Figure 3.2 shows the results of applying isotropic heat diffusion and TV filtering. Finally, unlike isotropic heat diffusion, ADF (in Subsection 2.2.1.4) does not apply smoothing the same way in different directions; thus, it is called anisotropic.

The performance of the pixel-based denoising methods (in Section 2.2.2) with the patch-based

(a) (b) (c)

(d) (e) (f)

Figure 3.1: Low-pass vs. BF: (a) an original *Lena* image, (b) an AWGN noisy image with $\sigma = 20$, (c) a denoised image using a 21×21 mean filter, (d) a denoised image using a 21×21 Gaussian filter $\sigma_d = 3.33$, (e) a denoised image using a 21×21 Yaroslavsky filter $\sigma_s = 0.5$, and (f) a denoised image using a 21×21, $\sigma_d = 3.33$ and $\sigma_r = 0.5$ BF. σ_d controls the space similarity for the blurring strength, and σ_r controls the intensity similarity for the edge preserving.

(a) (b)

(c) (d)

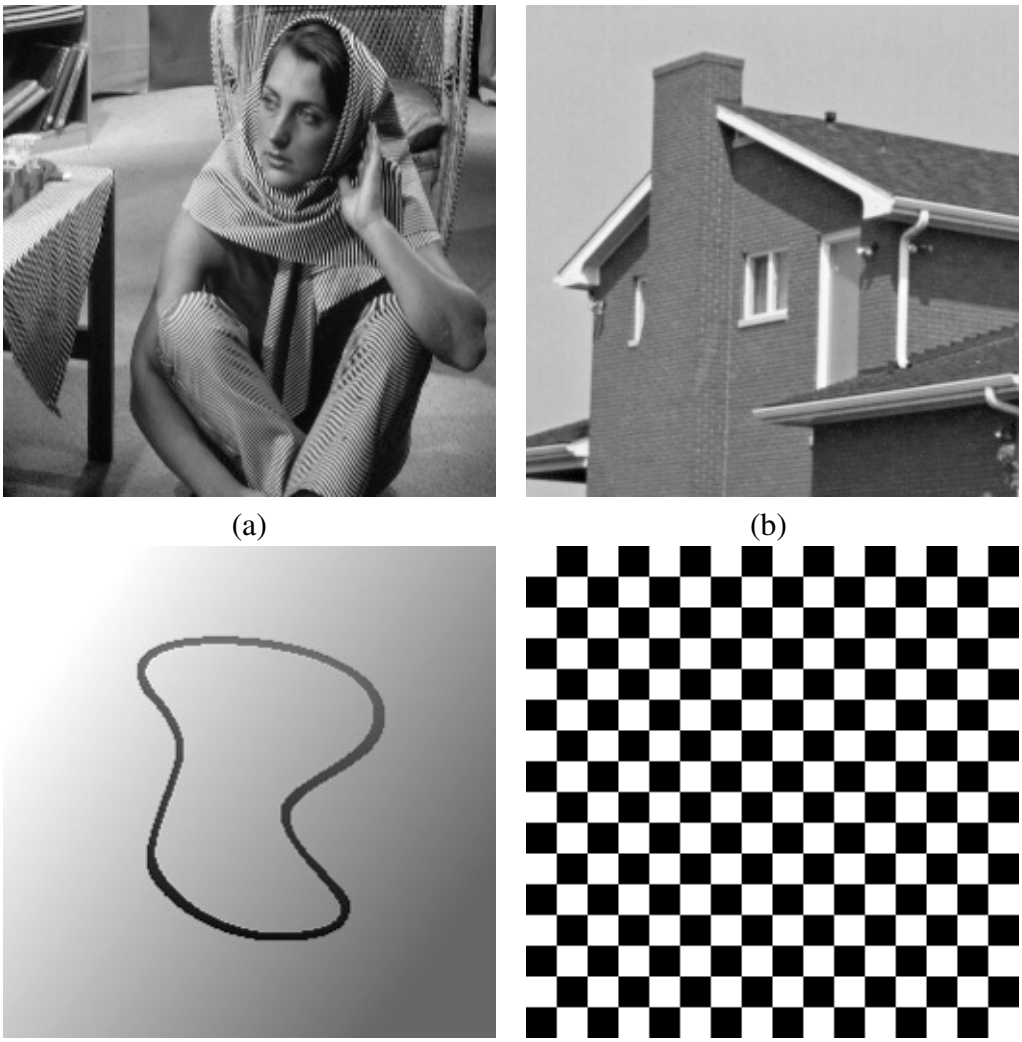
Figure 3.2: Isotropic heat diffusion and TV filtering: (a) an original *Lena* image, (b) an AWGN noisy image with $\sigma = 20$, (c) a denoised image using heat diffusion ($iteration = 50, \triangle t = 0.2$), and (d) a denoised image using a TV filter ($iteration = auto(121), \lambda = 0.02, \triangle t = 0.2$).

denoising methods for reducing additive noise at various noise levels have been experimentally studied. In addition, the issue of time consumption has been addressed.

Four images are used to run this experiment. The images have been chosen carefully to assist in distinguishing between the methods. The first two of the four images are natural scene images, `Barbara` and `House`; the other two are synthetic images, `CurvedBand` and `Chessboard`. The `Chessboard` image is a binary image while the other three images are grey-scale images. All four of the images are shown in Figure 3.3. The fine details in `Barbara` image helps in demonstrating how the various methods preserve the image clarity, whereas the sharp edges in the `House` image helps in demonstrating how various methods preserve edges. The grey gradations in `CurvedBand` image provide insight into the amount of smoothing that has been applied to images. The methods are also tested with the binary pattern repetitions in the `Chessboard` image.

The charts in Figure 3.4 show each method performance when $\sigma = 20$. By increasing the noise level, the contrast between methods becomes obvious unlike that seen when the noise level is low. Here BM3D (see Subsection 2.2.2.5) method achieved better results than were achieved with the other methods applied to all images. The patch-based denoising methods perform better than pixel-wise methods.

Figure 3.5 contains charts that illustrate the average execution time aspect for each image. The K-SVD (see Subsection 2.2.2.3) methods are very time-consuming. They are about ten times more expensive than any other method. Thus, K-SVD is excluded from the charts to make it easier to distinguish between methods. The NL-Means (see Subsection 2.2.2.1) comes second after the K-SVD. Although, the BM3D uses two stages to perform the denoising step, it is the fastest patch-based method. The time consumed for various values of noise is almost the same, so a high level of noise does not greatly affect these measurements. It is worth mentioning that the performance will be addressed in this thesis not complexity.



(a) (b)

(c) (d)

Figure 3.3: The four images used in the experiment: (a) *Barbara* image 512×512, (b) *House* image 256×256, (c) *CurvedBand* image 257×257, and (d) *Chessboard* image 256×256.

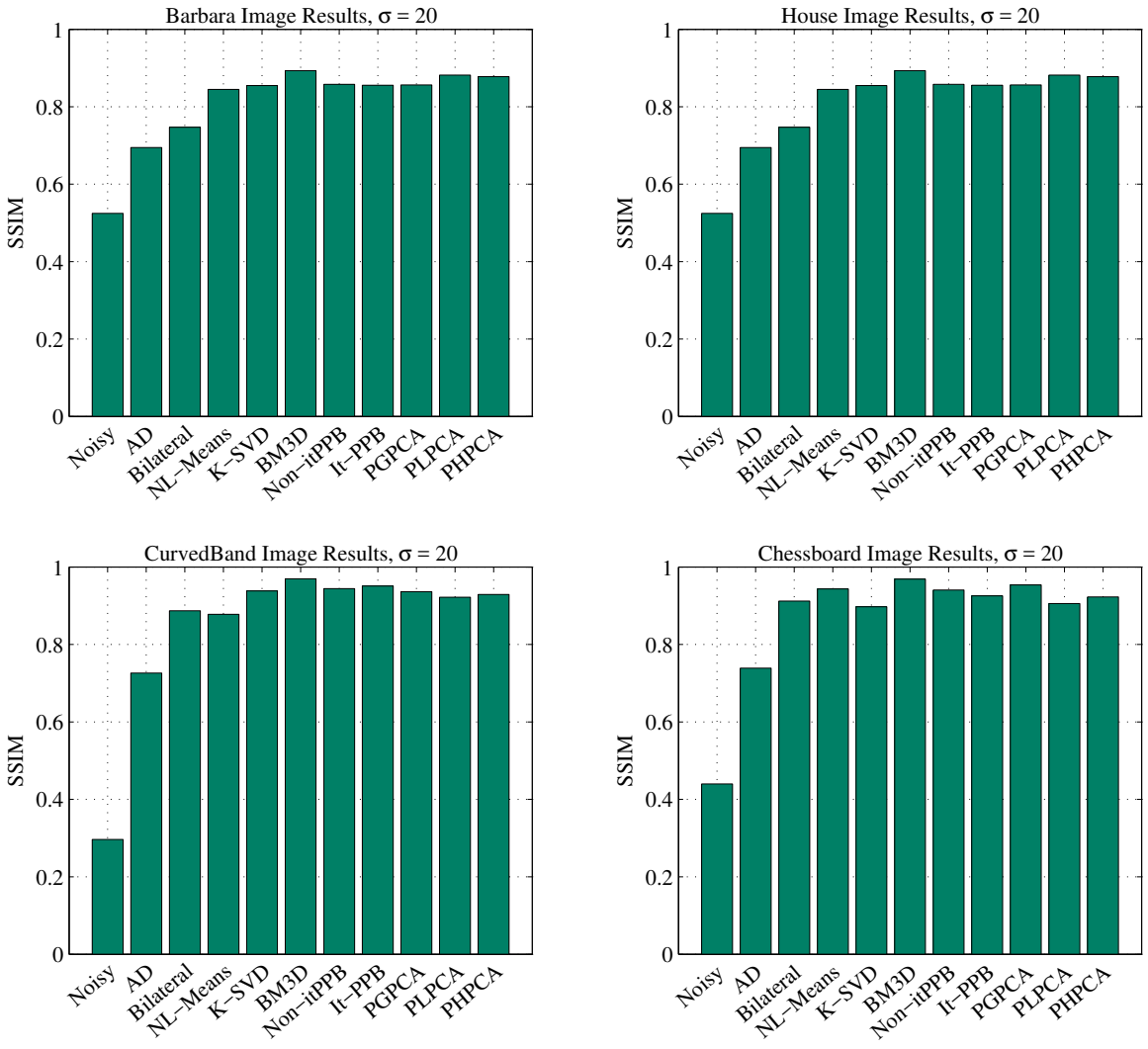


Figure 3.4: Charts summarize the performance of the denoising methods for the four images when the noise level is low ($\sigma = 20$). (The same graph is represented in tabular form in Appendix A)

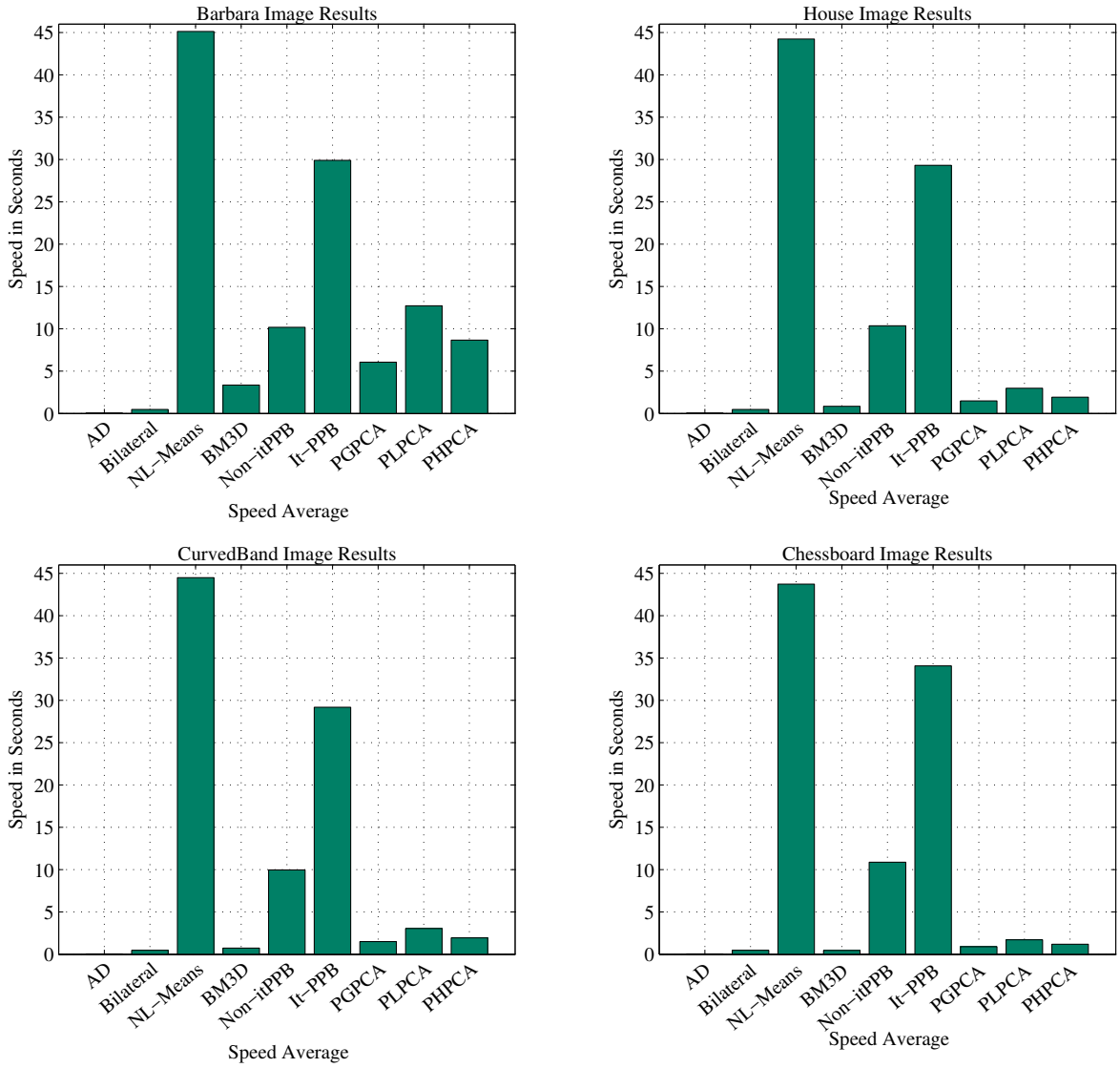


Figure 3.5: Charts showing the average time consumed in seconds for the various denoising methods excluding K-SVD. (The same graph is represented in tabular form in Appendix B)

Chapter 4

A New Patch-based Method for Single Image Denoising

In the previous chapter, we discussed various single view image denoising methods including the state-of-the-art patch-based denoising method (BM3D filter) of Dabov *et al.* [24]. This method preserves edges and blurs homogeneous areas by exploiting similarities among the various parts of the input image by using 3D transform-domain collaborative filtering, yet this denoising method has some performance limitations. This chapter presents a solution for the BM3D filter limitations and a modified single view image patch-based denoising method.

4.1 Idea of the Proposed Method

We will focus on the limitations of the fixed hard thresholding of the BM3D filter which was covered previously in Subsection 2.2.2 on page 40. Accordingly, several experiments were established to study this limitation. Here, we proposed a better solution for this fixed hard thresholding, and the use of a weighted average for the collaborative Wiener filtering step. The details of our solution is explained below.

Once a signal in the original BM3D is transferred to the frequency domain, the signal energy will be concentrated into a small range of coefficients. The signal energy coefficients values are high compared to the noise energy that is spread over a significant number of coefficients. Therefore, a threshold is used for the separation of the two energies, where the coefficients are passed through a threshold in order to remove all of the small coefficients. The original BM3D uses a fixed hard-

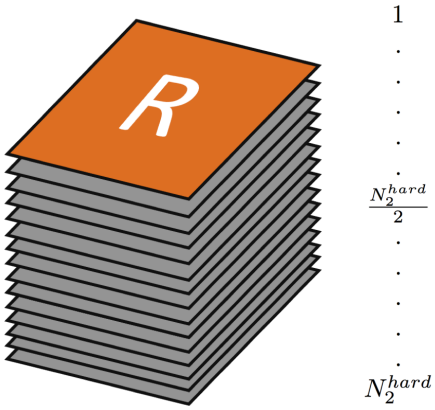


Figure 4.1: Sorting patches based on range weights: N_2^{hard} is the maximum number of similar patches.

thresholding operator as a shrinkage function for similar patches with a low luminance distance (see Subsection 2.2.2 on page 40). The fixed hard-thresholding operator γ^{3D} equals to $\lambda_{3D} \times \sigma$. $\lambda_{3D} = 2.7$ when $\sigma < 40$, or $\lambda_{3D} = 2.8$ when $\sigma \geq 40$. This fixed function blindly suppresses all of the transformed coefficients against a hard value, which is derived from all of the patches of the 3D array under the threshold, whether the coefficients are noise or not.

In our literature survey, we uncovered a pixel-based BF [95] that outperforms other low-pass filters. When we studied this filter, we found that geometric and luminance distances between pixels assist in suppressing random noise and preserve the fine details of noisy images. The idea of our proposed adaptive threshold is originally derived from this filter. The grouped patches "3D array" will be sorted based on the luminance distances by using a PSNR similarity measure, and then various thresholds are utilized based on a textures classification map. Figure 4.1 has a 3D array containing patches for the thresholding process. Patches with high weights are located at the top close to the reference patch location. If a patch is spatially located far from to the reference patch, more enforcement is applied when hard-thresholding the patches' transforming coefficients.

We ran various experiments to investigate whether using an adaptive thresholding operator that considers geometric and luminance distances between patches, could outperform the static thresholding operator. In [9], we have found that there was a slightly improvement achieved over using the fixed hard-thresholding of the original BM3D. Moreover, we extended the experiment by using various levels of thresholding on denoising two types of dataset images which include: texture images dataset, and flat images dataset. The datasets are shown in Figure 4.2 and 4.3.

After running the experiments on the images, we noticed that each of the dataset prefers a specific thresholding level for producing the best denoising results. By comparing our results with the original BM3D method, we have found that our thresholding levels assist in preserving greater

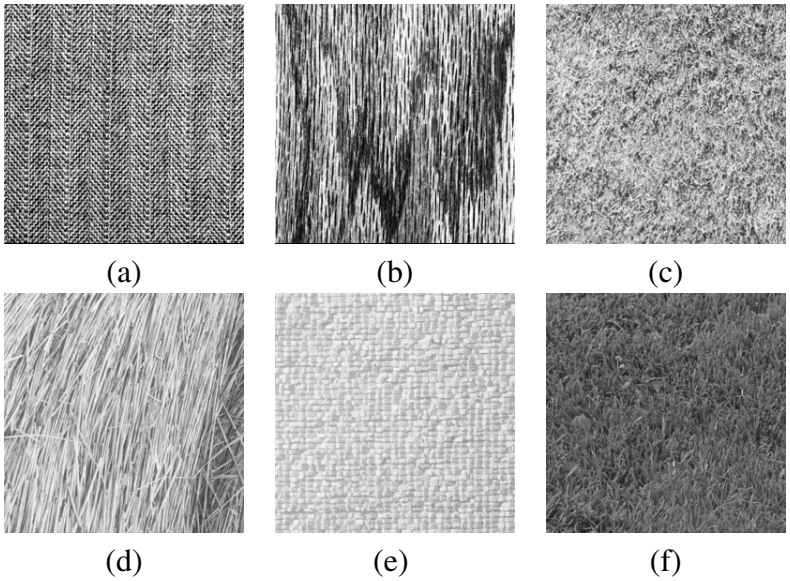


Figure 4.2: High textures image dataset: (a) `1.2.04.png` image 256×256, (b) `1.2.09.png` image 256×256, (c) `1.3.01.png` image 256×256, (d) 1.3.03.png image 256×256, (e) `1.3.10.png` image 256×256, and (f) `1.5.07.png` image 256×256. Dataset was obtained from [1].

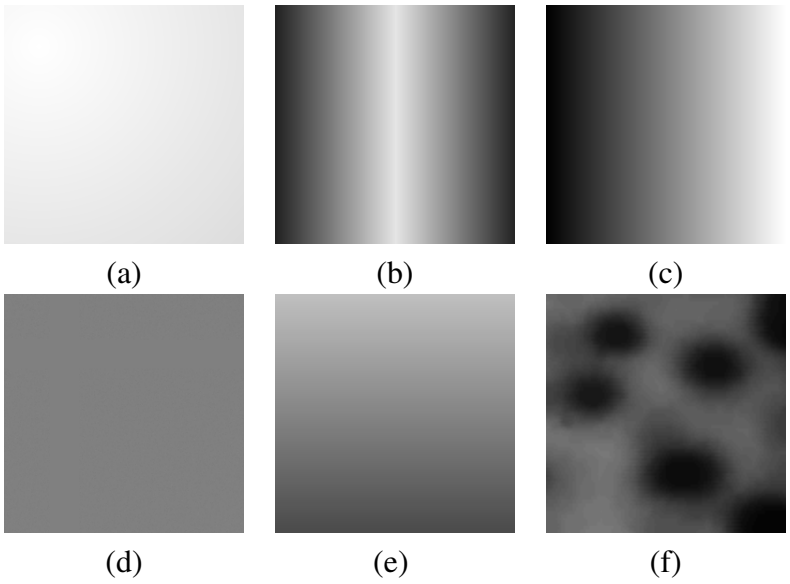


Figure 4.3: Flat image dataset: (a) `Fig0229(b).png` image 256×256, (b) `Fig1008(c).png` image 128×128, (c) `Fig0307(a).png` image 256×256, (d) `Fig0801(c).png` image 256×256, (e) `Fig1008(b).png` image 128×128, and (f) `Fig1056(a).png` image 256×256. Dataset was obtained from [2].

detail and the smoothing of properly flat regions. Result of the textures and flat regions are shown in Figure 4.4 and 4.5, respectively.

4.2 The Proposed Method

In this section, a *modified block matching 3D* filtering algorithm for denoising additive noisy images is proposed. A noise estimation phase was developed in the event the noise level was not specified. Figure 4.6 shows the modified filtering algorithm scheme. Elements of the proposed method are explained below in greater detail.

4.2.1 Adaptive Collaborative Thresholding

The first stage of our modified BM3D has an adaptive collaborative thresholding filter consisting of a classification map and a set of various thresholding levels and operators. A texture analysis method is utilized to create the classification map. Because the texture analysis method is a spatial domain approach, the classification map should be generated before transforming the noisy image. The classification map will provide a numeric value that represents the homogeneity of each patch. This value will be used later when choosing a suitable threshold level and operators for the thresholding of the patches group. This thresholding process estimates the true intensity of each patch. The general scheme for texture analysis and thresholding of this stage is illustrated in Figure 4.7.

4.2.1.1 Texture Analysis and the Classification Map

There are various texture analysis approaches. Some are simple; (e.g., standard deviation), and others are sophisticated; (e.g., statistical approaches: Laws' texture energy measures, *grey-level co-occurrence matrix* (GLCM)). Our method utilizes the statistical texture analysis approach GLCM. This approach was chosen over Laws' measures because it can be applied to various patch sizes. The GLCM measures texture areas globally, but we adapted it to be a local measure in order to fit our needs. The patch sizes of GLCM is adjusted according to the input patch size for our *modified block matching 3D* filter. GLCM in our method, first applies a BF and then calculates how often pairs of pixels with specific order $C_d(x, y)$ occur in each patch. Following this, a normalized co-occurrence matrix N_d is generated as:

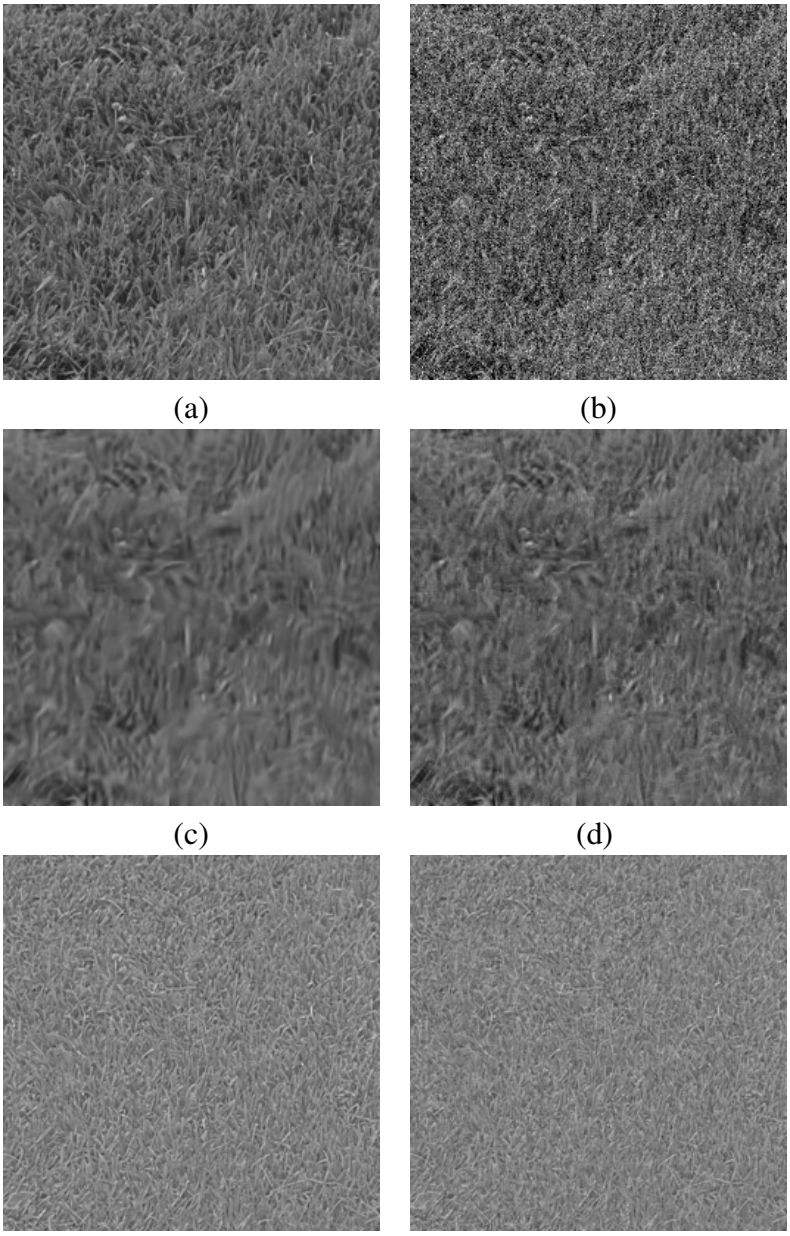


Figure 4.4: Denoising *texture* image by *modified* BM3D filter at(σ = 30): (a) original noiseless *texture* image, (b) a noise image σ = 30, (c) the denoised image (BM3D), (d) the denoised image (proposed method), (e) the residual image (f) = (b)-(c), and (f) the residual image (g) = (b)-(d).

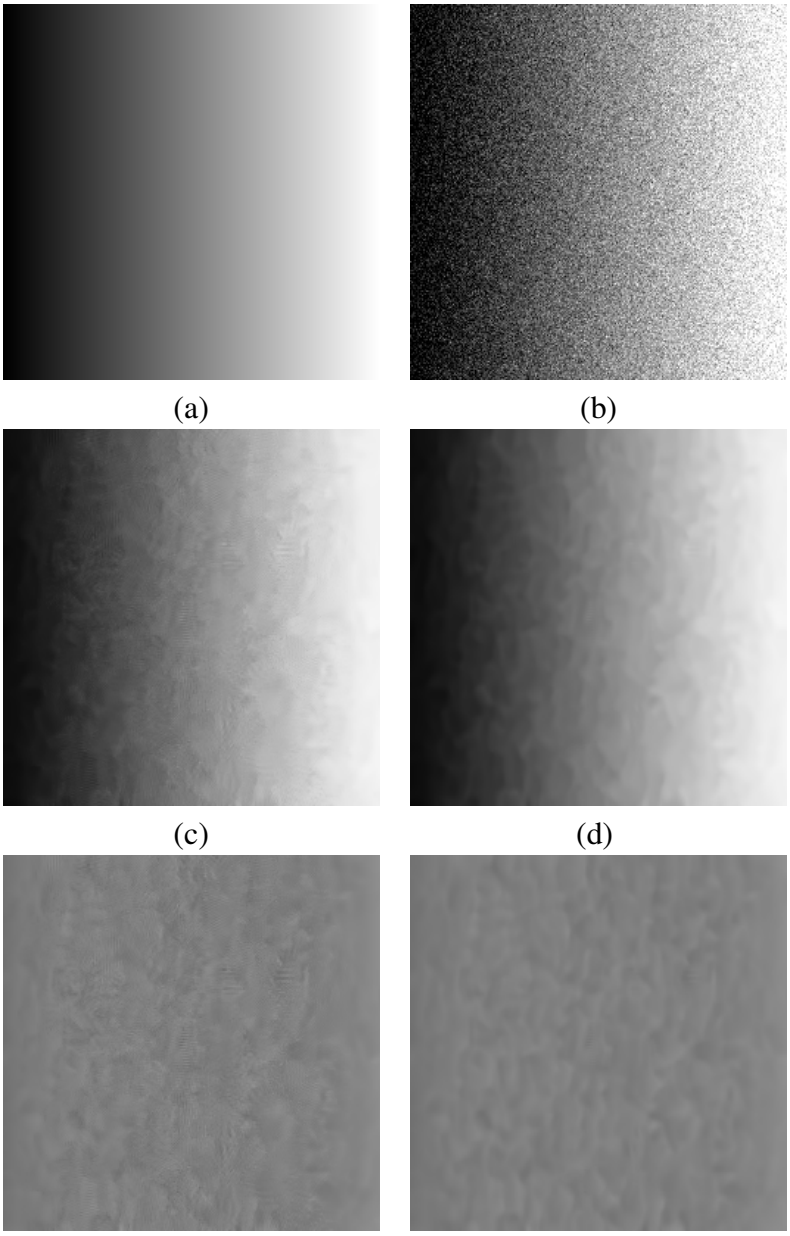


Figure 4.5: Denoising *flat* image by *modified* BM3D filter at(σ = 30): (a) original noiseless *flat* image, (b) a noise image σ = 30, (c) the denoised image (BM3D), (d) the denoised image (proposed method), (e) the residual image (f) = (b)-(c), and (f) the residual image (g) = (b)-(d).

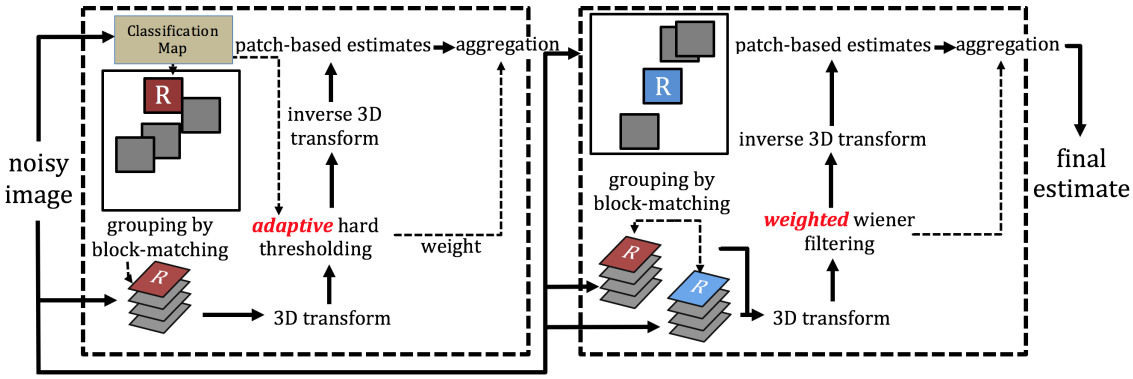


Figure 4.6: The two steps of the *modified block matching 3D* filtering scheme

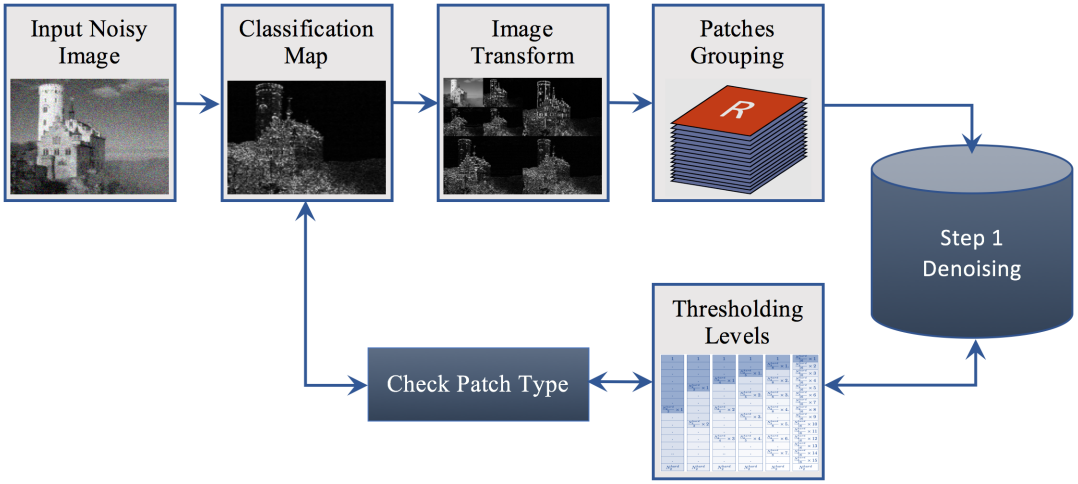


Figure 4.7: The general scheme for texture analysis and thresholding.

$$N_d(x,y) = \frac{C_d(x,y)}{\sum\limits_{x,y \in P} C_d(x,y)} \tag{4.1}$$

The following numeric feature, which represents homogeneity, is obtained from the normalized co-occurrence matrix of each patch:

$$Homogeneity = \sum_{x,y \in P} \frac{N_d(x,y)}{1+|x-y|}, \tag{4.2}$$

where x, y is the coordinates of each value. The output value of this equation ranges from (1.0 to 0.0). When the output value is close to 1, the patch is closer to a flat region. The homogeneity equation of each patch builds up the classification map. The classification map assists in distinguishing between patches with high structure and flat regions via a threshold value - this threshold is for texture analysis.

Empirically, we have found that the cut-off value varies based on two factors: the patch size and the noise level. For clarification, Figure 4.8 shows the effects of the thresholding values on classification maps. Figure 4.8 (a) shows the best threshold (0.1) when patch size is 2×2, Figure 4.8 (e) shows the best threshold (0.2) when patch size is 4×4, and Figure 4.8 (i) shows the best threshold (0.3) when patch size is 6×6. However, our classification map adjusts the thresholding values automatically for 8×8 or 12×12 patch sizes in case of $\sigma < 40$ $\sigma \geq 40$, respectively:

$$\begin{aligned}
&\text{In case of } 0 < \sigma < 40: \quad CMap_{I(x,y)} = \begin{cases} 255, & \text{if} \rightarrow \text{Homogeneity} > 0.25 \\ 0, & otherwise \end{cases} \\
&\text{In case of } 40 \leq \sigma \leq 100: \quad CMap_{I(x,y)} = \begin{cases} 255, & \text{if} \rightarrow \text{Homogeneity} > 0.15 \\ 0, & otherwise \end{cases}
\end{aligned} \tag{4.3}$$

where $CMap_{I(x,y)}$ is a pixel intensity of the classification map value at the coordinates (x, y). The thresholds have been chosen empirically.

(a) 2x2, $\tau = 0.1$ (b) 2x2, $\tau = 0.2$ (c) 2x2, $\tau = 0.3$

(d) 4x4, $\tau = 0.1$ (e) 4x4, $\tau = 0.2$ (f) 4x4, $\tau = 0.3$

(g) 6x6, $\tau = 0.1$ (h) 6x6, $\tau = 0.2$ (i) 6x6, $\tau = 0.3$

Figure 4.8: Classification maps of `Cameraman` image via various thresholds

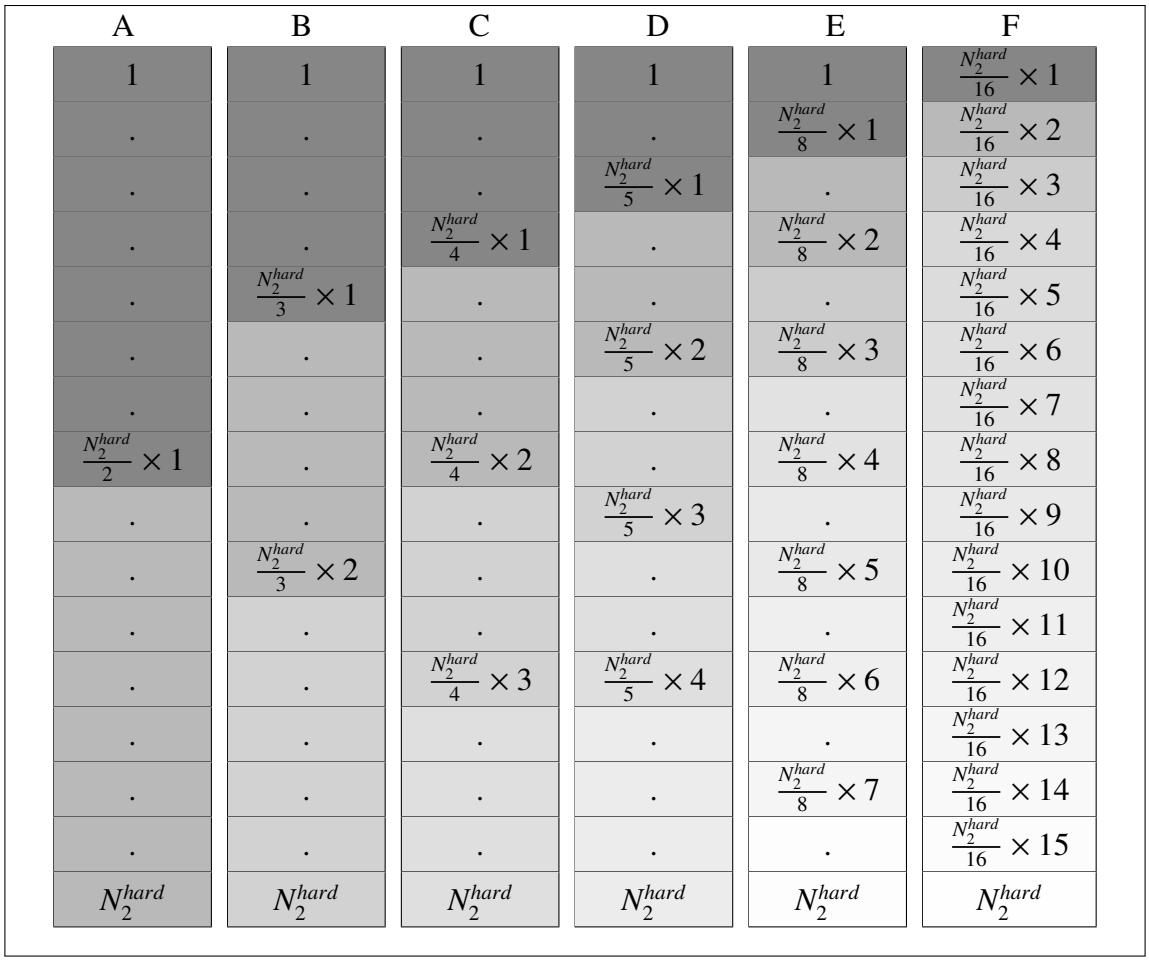


Figure 4.9: The six various thresholding levels: (Column A) 2 levels of thresholding, (Column B) 3 levels of thresholding, (Column C) 4 levels of thresholding, (Column D) 5 levels of thresholding, (Column E) 8 levels of thresholding, and (Column F) 16 levels of thresholding.

4.2.1.2 Thresholding Levels and Operators

In order to find the best thresholding levels and operators, we setup a battery of tests. This training stage had six various thresholding levels and operators. The overall thresholding and training procedure is illustrated in Figure 4.10. The thresholding levels are shown in Figure 4.9, and the operators values chosen were between (2.0-4.0). The two image datasets (texture images dataset [1] and flat images dataset [2]) were contaminated by AWGN with various noise levels (10-100) (σ). Each of the datasets were then denoised by using the various thresholding levels and operators, in order to find the best threshold. Finally, the best corresponding thresholding levels and operators for flat (non-structure) and textures (structure) images were recorded.

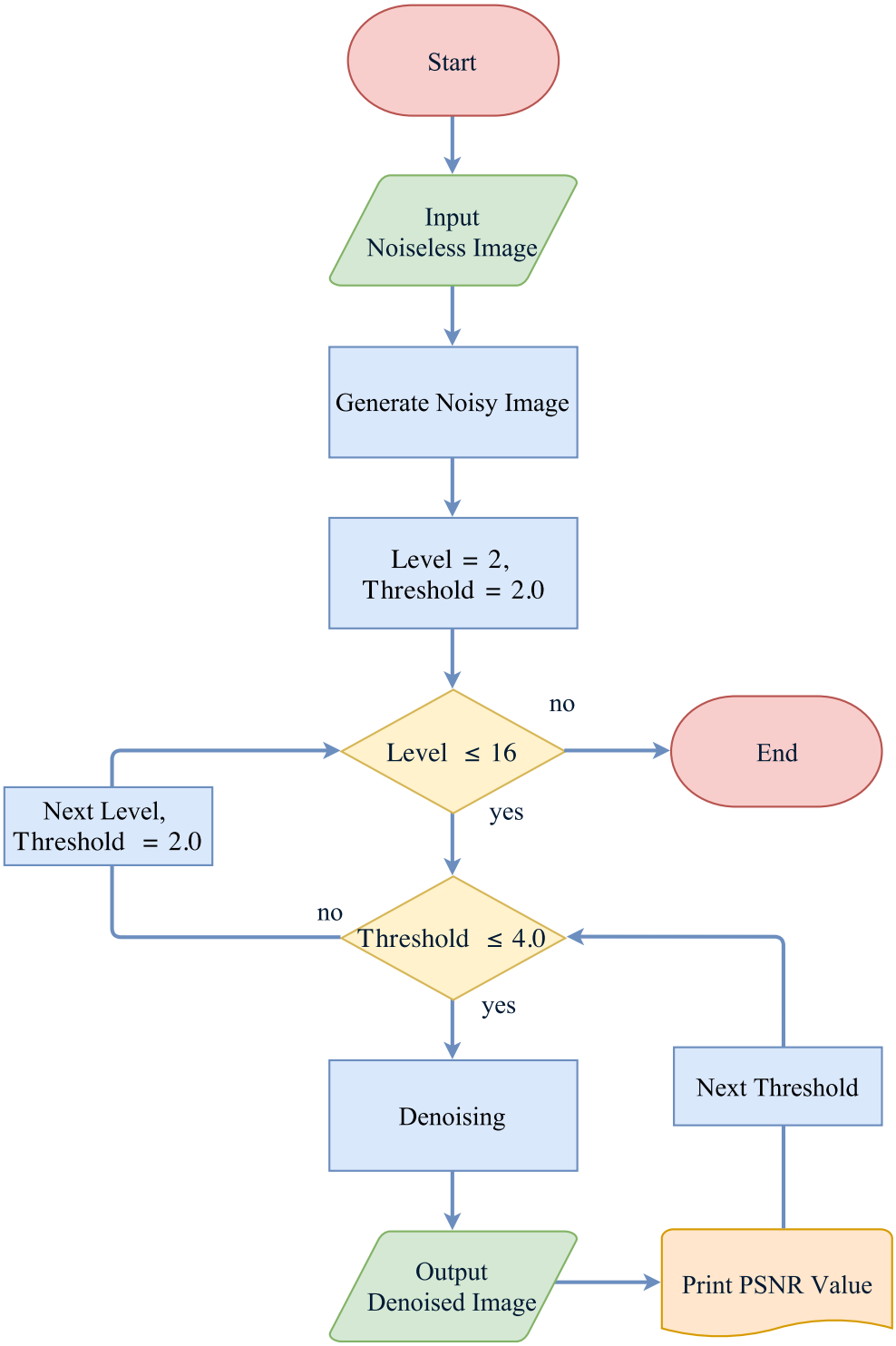


Figure 4.10: The overall thresholding and training procedure.

4.2.2 Modified Wiener Filtering Weights

Finally, we propose modified Wiener filtering. The Wiener shrinkage coefficients of the 3D array and the 3D transform coefficients of the noisy image are multiplied element-by-element. Before this process occurs, the Wiener shrinkage coefficients of the 3D array patches must be averaged. BM3D averages these coefficients arithmetically, but the modified Wiener filter uses weights for the 3D array patches when averaging; similar patches will also be assigned greater weight. Here PSNR is utilized as a patch similarity measure for assigning the weights. Finally, the inverse of the 3D transform is applied.

4.3 Tuning Parameters

4.3.1 Noise Range and Estimation

The original *block matching 3D* filtering algorithm assumes that the true additive noise power (standard deviation) is known and added as a parameter prior to the denoising procedure. Indeed, in real-life situations noise is unknown, and only the resulting noisy images exist. Users must subjectively estimate the true noise level in such situations. Incorrect estimation heavily degrades the result of the denoising algorithms, and it is always a challenge to precisely estimate the true noise levels of noisy images. Thus, a reliable noise level estimation process is strongly recommended in denoising algorithms.

In our proposed work, we considered adding a *noise level function* (NLF) as an initial stage for estimating noise levels before the actual denoising procedure in order to fit real-life situations. There exists spatial and frequency noise estimating techniques as perviously discussed. Since our denoising method is a frequency domain filter, using any of the frequency domain techniques with the proposed method is not a problem. Moreover, spatial domain estimation techniques can be applied prior to the transformation of the noisy image in the proposed method. In Section 2.4, some of spatial and frequency noise estimating techniques were deeply studied to choose the most suitable technique for the proposed method. From our experiments for the noise estimating techniques (in Subsection 2.4.3), we found that the noise level estimation based on PCA technique of Xinhao *et al.* [67] outperforms the other techniques, so we have chosen this technique to be embedded into our method when the noise level was not provided.

Table 4.1: Parameter set for our modified BM3D

Step	Symbol	Description	The Modified BM3D	
			$\sigma \leq 40$	$\sigma \geq 40$
Step 1 (hard) parameters	T^{2D}_{hard}	2D transformation	2D-Bior1.3	2D-DCT
	T^{3D}_{hard}	3D transformation	1D-Haar	1D-Haar
	N^{hard}_{1}	Patch size	8	12
	N^{hard}_{2}	3D array size	16	16
	N^{hard}_{step1}	Patch step size	3	4
	N^{hard}_{search}	Window size	39	39
	N^{hard}_{step2}	Window step size	1	1
	$N^{hard}_{Prev.}$	Small searching window	-	-
	β^{hard}	Kaiser window parameter	2.0	2.0
	λ_{3D}	3D thresholding operator	See Appendix C	
	T^{hard}_{match}	Similarity distance	2500	5000
Step 2 (Wiener) parameters	T^{2D}_{Wiener}	2D transformation	2D-DCT	2D-DCT
	T^{3D}_{Wiener}	3D transformation	1D-Haar	1D-Haar
	N^{Wiener}_{1}	Patch size	8	11
	N^{Wiener}_{2}	3D array size	32	32
	N^{Wiener}_{step1}	Patch step size	3	6
	N^{Wiener}_{search}	Window size	39	39
	N^{Wiener}_{step2}	Window step size	1	1
	$N^{Wiener}_{Prev.}$	Small searching window	-	-
	β^{Wiener}	Kaiser window parameter	2.0	2.0
	T^{Wiener}_{match}	Similarity distance	400	3500

4.3.2 Parameters

The parameters for our method were set similar to the original BM3D method. However, an adaptive threshold based on the geometric distance and patch energy (See table in Appendix C on page 138) replaces the hard-thresholding of the original BM3D (γ^{3D} is a hard-thresholding operator equal to $\lambda_{3D} \times \sigma$. $\lambda_{3D} = 2.7$ when $\sigma < 40$, or $\lambda_{3D} = 2.8$ when $\sigma \geq 40$). The parameters of our method are shown in Table 4.1.

4.3.3 Transforms Selections

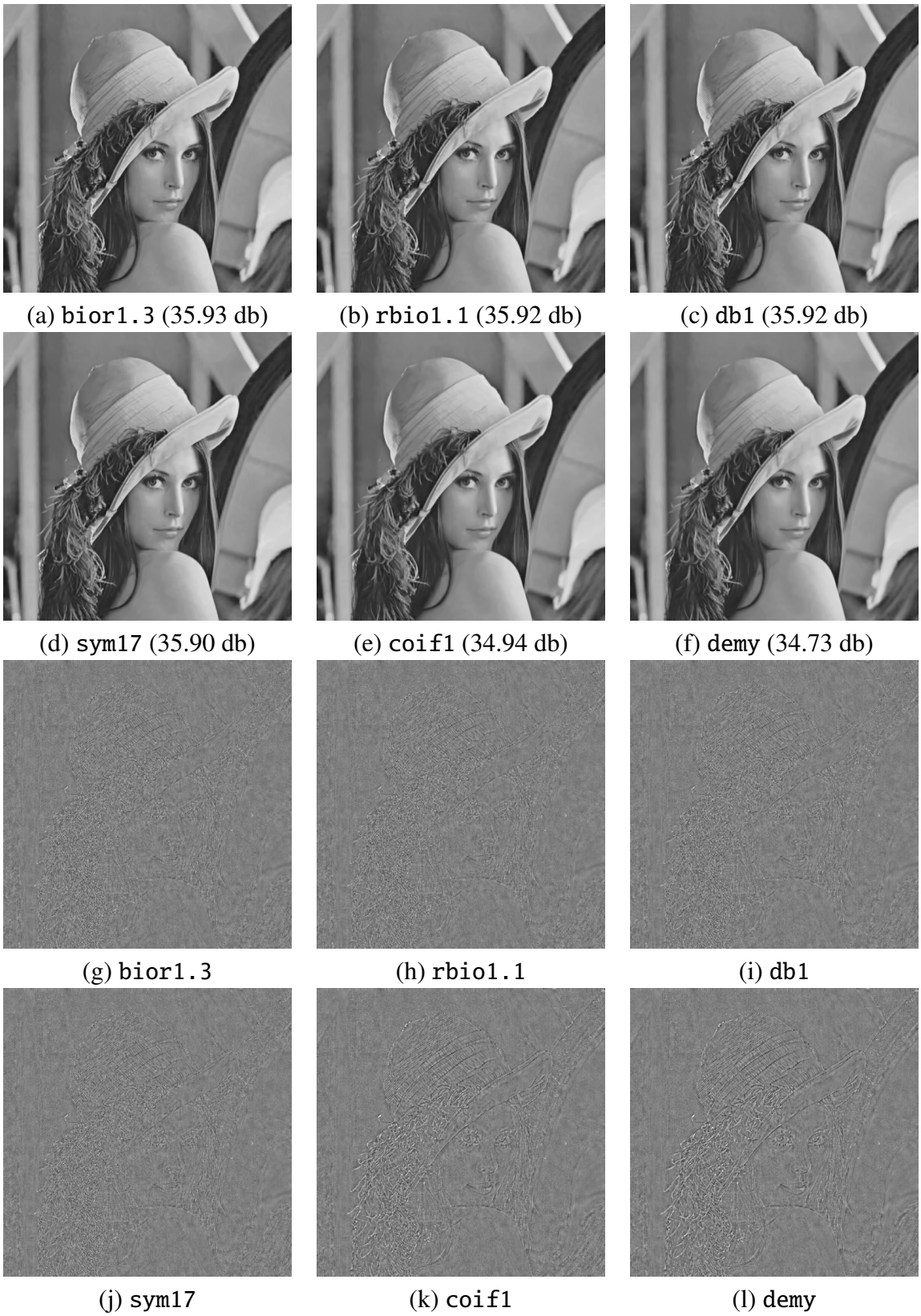
Filtering an image in the frequency domain involves three steps: first, the decomposition of a signal to obtain the transform coefficients; second, applying a filtering process to transform the coefficients, and third, reconstructing the coefficients to retrieve the signal.

Several families of discrete wavelet transforms were, (discussed in Section 2.6), utilized here in order to demonstrate the effects of the transformation on the filtering process. The various DWTs were examined for denoising the dataset of single images. The results show that `bior1.1`, `bior1.3`, `rbio1.1`, and `db1` wavelets came in on at the top of other wavelets. Thus, `bior1.3` was chosen to be applied in the proposed method.

The table in Appendix D on page 139 shows the performance of the *modified block matching 3D* filtering algorithm when denoising `Lena` image with various wavelet transforms. In order to investigate subjectively the differences between the various wavelet transforms, the results of using the wavelets for denoising `Lena` image at noise levels (σ = 10) and (σ = 30) are illustrated in Figures 4.11 and 4.12, respectively. Because of the large number of transforms used in this work (6 wavelet families with 76 wavelet transforms), the figures only show the best transform out of each wavelet family. The residual errors are shown in the figures as well.

4.4 Discussion

Suppressing more noise and preserving more detail in BM3D is possible when considering an adaptive hard-thresholding operator with a textures classification map. The weighted average assists in improving the BM3D collaborative Wiener filtering step. Various wavelet transforms can be utilized for the *modified block matching 3D* filtering method, but `bior1.1`, `bior1.3`, `rbio1.1`, and `db1` wavelets are the most suitable wavelets. The performance of the *modified block matching 3D* filtering method will be discussed later in Chapter 6.



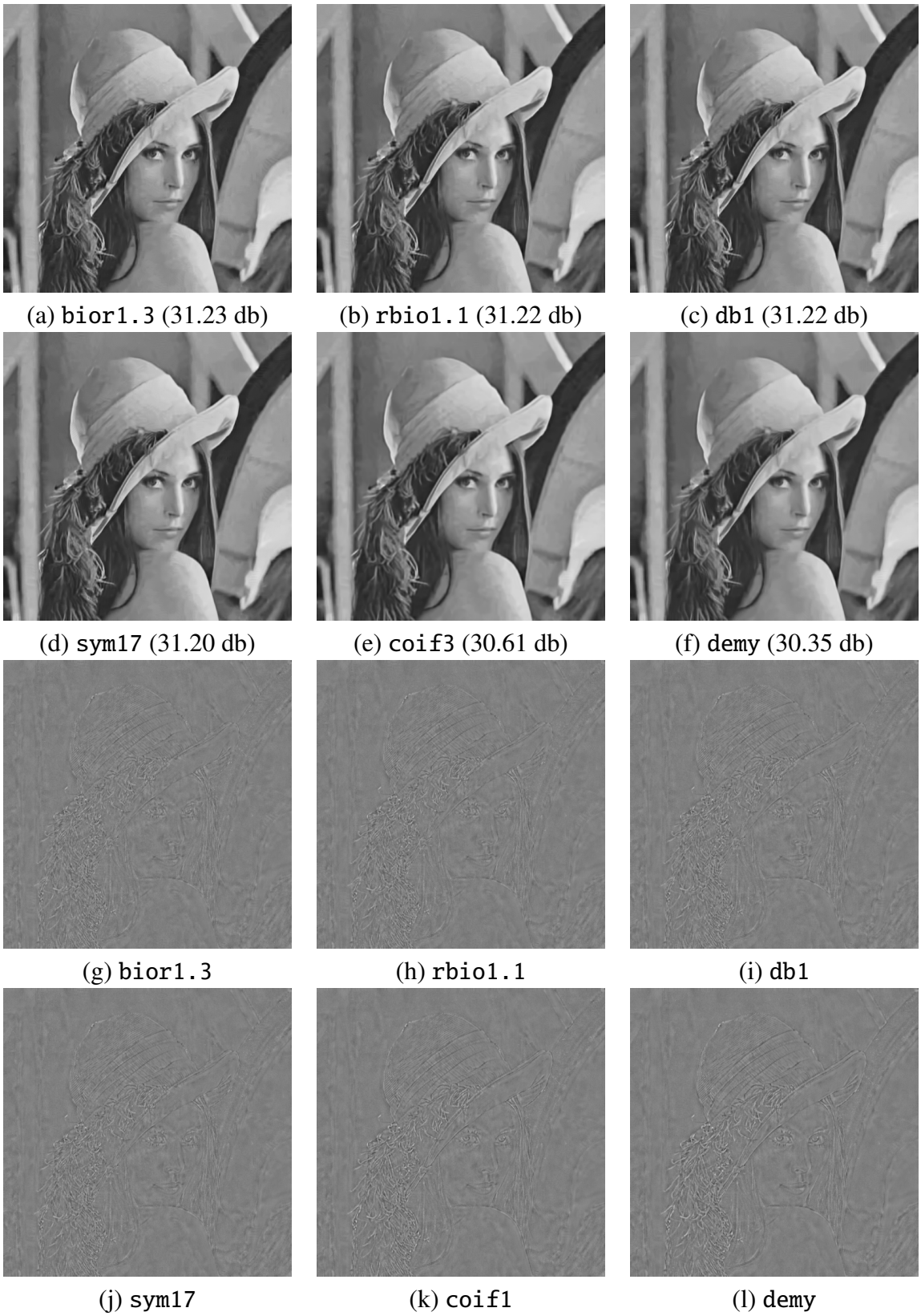
(a) bior1.3 (35.93 db) (b) rbio1.1 (35.92 db) (c) db1 (35.92 db)

(d) sym17 (35.90 db) (e) coif1 (34.94 db) (f) demy (34.73 db)

(g) bior1.3 (h) rbio1.1 (i) db1

(j) sym17 (k) coif1 (l) demy

Figure 4.11: Denoising *lena* image by *modified block matching 3D* filter with various wavelets at(σ = 10): (a) bior1.3, (b) rbio1.1, (c) db1, (d) sym17, (e) coif1, (f) demy, (g),(h),(i),(j),(k), and (l) are residuals.



(a) bior1.3 (31.23 db) (b) rbio1.1 (31.22 db) (c) db1 (31.22 db)

(d) sym17 (31.20 db) (e) coif3 (30.61 db) (f) demy (30.35 db)

(g) bior1.3 (h) rbio1.1 (i) db1

(j) sym17 (k) coif1 (l) demy

Figure 4.12: Denoising *lena* image by *modified block matching 3D* filter with various wavelets at(σ = 30): (a) bior1.3, (b) rbio1.1, (c) db1, (d) sym17, (e) coif1, (f) demy, (g),(h),(i),(j),(k), and (l) are residuals.

Chapter 5

New Patch-based Methods for Multi-view Image Denoising

Redundancy and similarity among the various parts of multi-view images assist in enhancing the image denoising process. Here efficient patch similarity assessments play a significant role in maximizing the number of similar patches in order to reach an estimate of the pixels' real values. Those similarity metrics include: the SSIM index, the MD, and the SVD-based similarity assessments. NL-Means filtering, is a simple and spatial domain patch-based filter, which could be adapted along with reliable similarity measures in order to produce an effective enhancing approach that preserves edges and blurs the homogeneous areas. This chapter presents new patch-based image filtering approaches for additive noise reduction in noisy multi-view images. We assume that both the multi-view images are contaminated with the same noise levels. In Section 5.1, the adaptation of a NL-Means filtering algorithm for multi-view image denoising is introduced. In Section 5.2, a new multi-view image denoising approach based on NL-Means with a SSIM index is discussed. In Section 5.3, a new approach for denoising a multi-view image using NL-Means along with MD and SVD-based similarity assessments is presented.

5.1 Non-local Means for Multi-view Image Denoising

In this section, the utilization of Buades *et al.* [17] patch-based method for stereoscopic image denoising is presented. The novelty of this method is its use for filtering multi-view images along with better quality assessment. After extending the search windows of the NL-Means to cover the

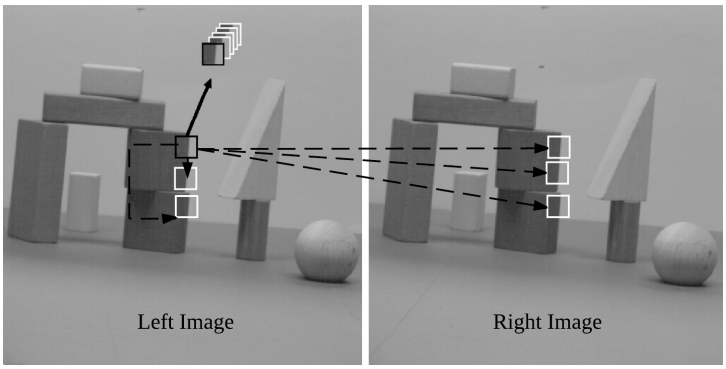


Figure 5.1: Collecting similar patches from a stereoscopic image: the patch with a black border is the reference patch, and the patches with the white borders are similar patches.

left and right portions of a stereoscopic image, the larger number of patches will provide estimates of the reference patches. Figure 5.1 shows how the number of similar patches can be increased.

The estimated value $NLMeans\,[vl]_k$, for each pixel on the left image at location k, is computed as:

$$NLMeans\,[vl]_k = \sum_{j \in I} \omega(k, j)\,[v]_j\,, \tag{5.1}$$

where $[vl]_k$ and $[v]_j$ are the pixel intensities of the left image at location k and j, respectively. Here j could be located in either the left or the right image. And $\omega(k, j)$ is a similarity measure between the pixels of k and j.

The similarity weight measure satisfies the condition $0 \leq \omega(k, j) \leq 1$ and $\sum_j \omega(k, j) = 1$. It also depends on the grey level similarity and the Euclidean distance between $N\,[vl]_k$ and $N\,[v]_j$, where $N\,[vl]_k$ and $N\,[v]_j$ denote the square neighbourhoods of a fixed size, centred at pixel k and j, respectively. The weights are described as:

$$\omega(k, j) = \frac{1}{Z(k)} e^{-\frac{\|(N[vl]_k) - (N[v]_j)\|^2}{h^2}}, \tag{5.2}$$

where $Z(k)$ is a normalization factor and h is a filtering parameter set depending on the noise level.

5.2 Non-local Means with Structural Similarity

Patch-based denoising methods achieve better results when there are enough similar patches that are accurately grouped before starting the denoising process. Thus, choosing an accurate similarity assessment can improve the results of the whole denoising process. The main contribution of this section is the use of SSIM with an extended search area, which makes this approach better for use with similar patches from both the left and right images.

SSIM and NL-Means [7] are further discussed in order to improve the method of extracting depth information and creating disparity maps of the stereoscopic images. The SSIM is then utilized to identify the locations of similar patches from within the input images. The NL-Means algorithm is then adopted to denoise the collected patches originating from the input images.

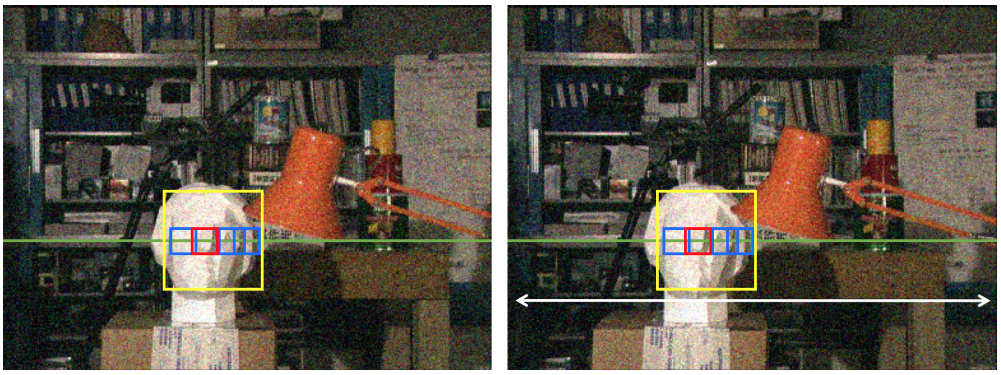
The use of the algorithm is validated in denoising various stereoscopic images at various noise levels in the following chapter. The experimental results show that the denoising performance of the algorithm is better than the original NL-Means and the *stereoscopic denoising* with MSE (Stereo-MSE) [49] methods at low noise levels ($\sigma \leqslant 20$).

5.2.1 Structural Similarity Index

The SSIM index is a metric used for measuring the similarity between patches. Unlike the traditional approaches (e.g., PSNR and MSE), SSIM has been proven to be consistent with human perception. This index takes into consideration image degradation as a perceived change in structural information, whereas traditional approaches only estimate perceived errors in the image data. The SSIM assessment's value of two patches of size $n \times n$ is calculated as:

$$SSIM(N[vl]_k, N[vr]_q) = \frac{(2\mu_L\mu_R + C_1)(2\sigma_{LR} + C_2)}{\left(\mu_L^2 + \mu_R^2 + C_1\right)\left(\sigma_L^2 + \sigma_R^2 + C_2\right)}, \tag{5.3}$$

where $N[vl]_k$ is a reference patch from the left image, $N[vr]_q$ is the corresponding patch from the right image, μ_L and μ_R are the means of the reference and its corresponding patches, respectively. Here σ_L^2 and σ_R^2 are the variances of the reference and the corresponding patches, respectively. And σ_{LR} is the covariance between the reference and the corresponding patches. Finally, C_1 and C_2 are the constants used to avoid instability.



left image right image

Figure 5.2: A full search for a row t: at each pixel of row t_{left} from the left stereoscopic image, full search will be employed for the row t from the right image t_{right}.

5.2.2 Full Searching

At each pixel of a row t_{left} from the left stereoscopic image, the algorithm employs a full search for row t_{right} from the right image for identifying a similar search window's location. Figure 5.2 illustrates the full searching window process between multi-view images before grouping similar patches.

5.2.3 Algorithm Outline

At each pixel k, a reference patch $N[vl]_k$ from the left image is centred at location k and its search window is obtained. The right image will then be searched, in raster scan fashion, for a similar patch $N[vr]_q$ centred at location q using the structural similarity $SSIM\left(N[vl]_k, N[vr]_q\right)$ in order to identify the location of a similar search window. Similar patches from the two windows will be grouped and assigned weights ω; higher weights will be assigned to patches with the highest level of similarity to the reference patch. The true pixel of the left image value is then estimated by taking the weighted average of the patches of the windows $NLMeans[vl]_k$. The scheme of SSIM and NL-Means is illustrated in Figure 5.3. The pseudocode of this algorithm is shown in Algorithm 5.1.

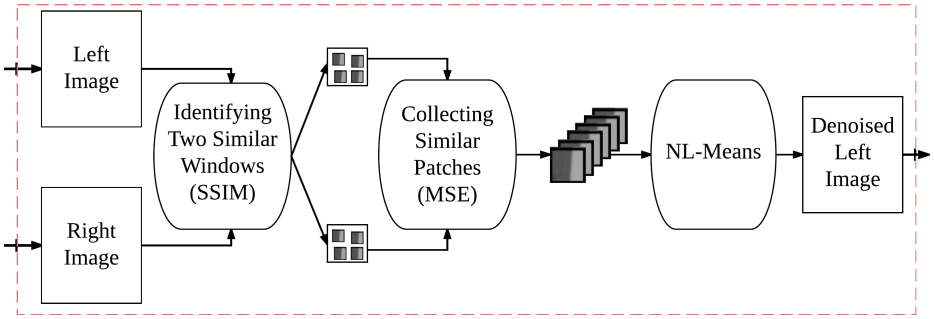


Figure 5.3: A block diagram of the structural similarity assessment with the NL-Means for stereoscopic image denoising

Algorithm 5.1 Denoising stereoscopic image using structural similarity with NL-Means

Require: stereoscopic image-left and right images as a input
Ensure: images dimensions must be equal
Output: left images
$lw \leftarrow$ left image width
$rw \leftarrow$ right image width
for $(k = 0;\ k < lw;\ k++)$ **do**
  $N[vl]_k \leftarrow$ obtain from the left image patch $n \times n$ centred at location k
  $Wl_k \leftarrow$ searching window $M \times M$ centred at location k
  for $(q = 0;\ q < rw;\ q++)$ **do**
    $N[vr]_q \leftarrow$ obtain from the right image patch $n \times n$ centred at location q
    if $(SSIM\left(N[vl]_k, N[vr]_q\right) > distance)$ **then**
      distance = $SSIM\left(N[vl]_k, N[vr]_q\right)$
      j = q
    end if
    $Wr_j \leftarrow$ searching window $M \times M$ centred at location j
  end for
  $maxP \leftarrow$ maximum number of similar patches
  $mp = 0$
  while $(mp < maxP)$ **do**
    $\omega(k, mp)$ ⇠ assign weights to each patch from windows Wl_k and Wr_j
    $NLMeans[vl]_k$, weighted average patches of Wl_k and Wr_j
    $mp++$
  end while
end for

5.3 Non-local Means with Differences Patch Similarity

In the previous work of Section 5.2 [7], the NL-Means method was adapted for filtering stereoscopic images in order to improve the applications of multi-view images. Yet, the previous work failed to achieve encouraging results when denoising the stereoscopic images with high noise levels ($\sigma > 20$) because SSIM was utilized. This is due to the fact that when the noise level increases, SSIM favours those patches with a similar noise pattern not the actual structure of the patches. However, in this section, a new stereoscopic image denoising algorithm, similar to the previous approach, is presented.

The algorithm takes a stereoscopic image as an input. Then, either the MD or the SVD-based (SVD-based) similarity assessments is utilized for obtaining better locations for the similar searching windows in the input images. Next, the NL-Means algorithm is adapted for denoising the patches within the bounded search areas. A summary of the contributions of this subsection's work in comparison to the previous work is as follows:

1. A SVD-based patch similarity metric is employed instead of the SSIM as a visual quality assessment.
2. A MD patch similarity metric as a quality assessment is employed for multi-view image denoising.
3. A bounded search method is utilized instead of the full search method.
4. The outputs contain two images: the left and the right.

The experimental results presented in the following chapter show that this algorithm outperforms both the original NL-Means and our previous approach to stereoscopic image denoising using NL-Means with *structural similarity* (S-SSIM). And it helps to create more accurate disparity maps at the various noise levels.

5.3.1 Patch Similarity Assessments

Generally, patch-based denoising methods rely heavily on accurate patch similarity assessments. Those similarity assessments measure the similarity between patches based on the apparent differences. This section proposes using more suitable similarity metrics including: SVD-based similarity and MD assessments. The SVD-based measure utilizes SVD to assist in identifying the

similarity between the reference patch and other patches. Meanwhile, The MD is used to determine the similarity between patches by computing the absolute maximum difference between the patches.

5.3.1.1 SVD-based Similarity Metric in Patch Similarity Assessment

The HVS is sensitive to structural, luminance, and textural changes in images, so these three changes must be measured in producing an effective visual quality assessment. The structural image similarity index assesses these changes in the real domain, and the result is an efficient visual quality assessment. However, this measure is highly unsuitable for measuring noisy images. As an alternative approach to overcoming the limitations of SSIM, Narwaria *et al.* presented the SVD-based image visual quality assessment robust against noise [79]. Indeed, what makes their assessment tool more effective against noise is that they utilize SVD as a transformation before measuring the changes. The transformation assists in distinguishing between noise and signals, when measuring the changes. This SVD-based similarity assessment consists of two stages: in the first stage, the visual quality features are extracted from the patches and their singular values are calculated, whereas a machine learning scheme is utilized to identify the most discriminant features in the second stage. The large value of an SVD-based assessment means that the patch is of poor quality.

5.3.1.2 Absolute Maximum Difference as a Patch Similarity Assessment

MD assessment computes the absolute maximum difference in order to determine the similarity between patches. This method is simpler and faster than the SVD-based similarity assessment. The lower the value of MD, the better the quality of the patch. The lower value of the MD, the better the quality of the patch. The MD is defined as:

$$MD = \max\left(\left|N\left[vl\right]_k - N\left[vr\right]_q\right|\right) \tag{5.4}$$

5.3.2 Bounded Searching

At each pixel i of row t_{left} from the left stereoscopic image, the algorithm employs a bounded search $[i-\rho, i+\rho]$ in row t_{right} from the right image to identify a similar searching window, where

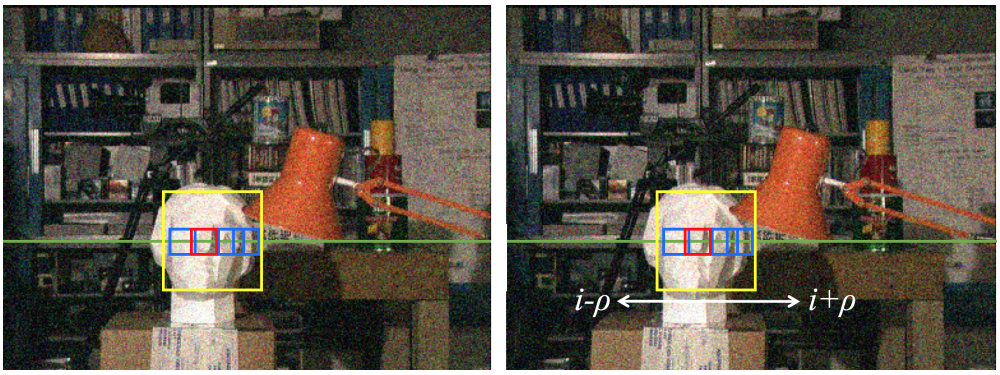


Figure 5.4: Bounded search for a row t: at each pixel i of row t_{left} from the left stereoscopic image; the bounded search $[i-\rho, i+\rho]$ will be employed for a row t from the right image t_{right}. Theρ adjusts the searching size.

ρ is an operator that adjusts the search size. Figure 5.4 illustrates the bounded searching process.

5.3.3 Algorithm Outline

At each pixel k_{left} and k_{right}, in both the left and right images, there are the reference patches $N[vl]_k$ and $N[vr]_k$ centred at location k with their searching windows. Those pixels will be estimated based on their weight by averaging the similar patches that are located within the search windows. Here, grouping similar patches from both of the search windows is the next required stage. Whereas either the MD or the SVD-based similarity metrics will be utilized in order to find the best match for the reference patch within a bounded searching area in the right image. After that, similar patches from the two windows will be grouped, and weights ω will be assigned to each of the grouped patches. Patches, which are similar to the reference patch, are assigned higher weights. The weights are assigned as described in Equation 5.2. Patches will be average weighted in order to estimate the true pixels of the left and right images. The estimated values are computed as described in Equation 5.1. The algorithm is illustrated in Figure 5.5. The algorithmic steps of this approach are shown in Algorithm 5.2.

Algorithm 5.2 Denoising stereoscopic image using MD or SVD-based similarity with NL-Means

Require: stereoscopic image-left and right images as a input
Ensure: images dimensions must be equal
  lw,$rw \leftarrow$ left and right image width
  $\rho \leftarrow$ searching size parameter
  for $(k = 0;\ k < lw;\ k++)$ **do**
    $N[vl]_k \leftarrow$ obtain from the left image patch $n \times n$ centred at location k
    $Wl_k \leftarrow$ searching window $M \times M$ centred at location k
    for $(q = k - \rho;\ q < \rho;\ q++)$ **do**
      $N[vr]_q \leftarrow$ obtain from the right image patch $n \times n$ centred at location q
      if $(Dist\left(N[vl]_k, N[vr]_q\right) < distance)$ **then**
        $distance = Dist\left(N[vl]_k, N[vr]_q\right),\ j = q$
      end if
      $Wr_j \leftarrow$ searching window $M \times M$ centred at location j
    end for
    $maxP \leftarrow$ maximum number of similar patches
    $mp = 0$
    while $(mp < maxP)$ **do**
      $\omega(k, mp) \leftarrow$ assign weights to each patch from windows Wl_k and Wr_j
      $NLMeans[vl]_k$, weighted average patches of Wl_i and Wr_q
      $mp++$
    end while
  end for
  for $(q = 0;\ q < rw;\ q++)$ **do**
    $N[vr]_q \leftarrow$ obtain from the left image patch $n \times n$ centred at location q
    $Wr_q \leftarrow$ searching window $M \times M$ centred at location q
    for $(k = q - \rho;\ k < \rho;\ k++)$ **do**
      $N[vl]_k \leftarrow$ obtain from the right image patch $n \times n$ centred at location k
      if $(Dist\left(N[vr]_q, N[vl]_k\right) < distance)$ **then**
        $distance = Dist\left(N[vr]_q, N[vl]_k\right),\ i = k$
      end if
      $Wl_i \leftarrow$ searching window $M \times M$ centred at location i
    end for
    $maxP \leftarrow$ maximum number of similar patches
    $mp = 0$
    while $(mp < maxP)$ **do**
      $\omega(q, mp) \leftarrow$ assign weights to each patch from windows Wr_j and Wl_k
      $NLMeans[vr]_q$, weighted average patches of Wr_q and Wl_i
      $mp++$
    end while
  end for

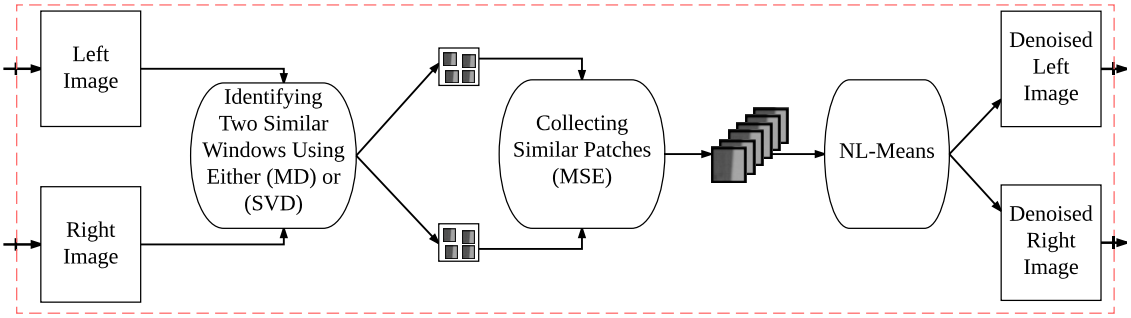


Figure 5.5: A block digram for stereoscopic image denoising using MD or SVD-based similarity assessments with NL-Means

Chapter 6

Results and Discussion

The objective of this chapter is to experimentally study the performance of the proposed patch-based methods for single and multi-view image denoising. The methods are evaluated both quantitatively and qualitatively. In the quantitative evaluation, this thesis uses full reference similarity measures - in other words the reference image, which is assumed to be a constant and free of noise (a true image), is compared with the denoised image. While `C++` code is implemented in the patch-based single image denoising method, `MatLab` code is also employed in the patch-based multi-view image denoising methods. The computer processor used is an Intel® Core^{TM} i7 (2.5 GHz). In Section 6.1, various image similarity measures for evaluating the denoising methods are addressed. In Section 6.2, the patch-based single image denoising method is evaluated. In Section 6.3, the patch-based multi-view image denoising methods evaluations are considered.

6.1 Image Similarity Measurements

Two image similarity metrics are explored as objective assessments for assessing the overall quality of the denoised images: (1) the MSSIM, and (2) the PSNR. The poorest result for MSSIM is zero, and the best result is 1. The higher the PSNR values, the better the results. The *mean structural similarity* evaluates images as:

$$SSIM\left(u\left(x,y\right),\hat{u}\left(x,y\right)\right)=\frac{\left(2\mu_{u(x,y)}\mu_{\hat{u}(x,y)}+C_1\right)\left(2\sigma_{u(x,y)\hat{u}(x,y)}+C_2\right)}{\left(\mu^2_{u(x,y)}+\mu^2_{\hat{u}(x,y)}+C_1\right)\left(\sigma^2_{u(x,y)}+\sigma^2_{\hat{u}(x,y)}+C_2\right)}, \tag{6.1}$$

$$MSSIM\left(u\left(x,y\right),\hat{u}\left(x,y\right)\right)=\frac{1}{M\times N}\sum_{x=1}^{M}\sum_{y=1}^{N}SSIM\left(u\left(x,y\right),\hat{u}\left(x,y\right)\right), \tag{6.2}$$

where $\mu_{u(x,y)}$ and $\mu_{\hat{u}(x,y)}$ are the mean of the true reference image patch and the noisy image patch, respectively, $\sigma_{u(x,y)}$ and $\sigma_{\hat{u}(x,y)}$ are the standard deviations of each image, $\sigma_{u(x,y)\hat{u}(x,y)}$ represents the covariance of the two images, and C_1, C_2, and C_3 are the constants used to avoid instability. The PSNR is defined as:

$$PSNR=10\log_{10}\left(\frac{(2^n-1)^2}{\text{MSE}}\right) \tag{6.3}$$

where n is an integer number representing the number of bits per pixel, $n = 8$ in the case of grey-scale images.

6.2 New Single Image Denoising Evaluation

In this section our *modified* BM3D filtering method at various noise levels is discussed. The original BM3D parameters are used for the *modified* BM3D filtering method. Ten grey-scale images were used during this experiment. The images are shown in Figure 6.1. In Subsection 6.2.1 and Subsection 6.2.2, the methods are evaluated both quantitatively and qualitatively, respectively.

6.2.1 Quantitative Evaluation

The results of the *modified* BM3D filtering method are shown in Table 6.1, which compares the performance of that method with that of the original BM3D. In the table, each noise level (σ) has two rows: the first row shows the results of the original BM3D, and the second row shows the results of the *modified* BM3D filtering method. PSNR was utilized as a similarity metric to objectively assist the deference between the original and denoised images. The higher PSNR values are highlighted with a bold font. The results are computed by measuring the differences between the original noise-free images and the denoised ones. From the table, the level of our method performance is higher than that of the original BM3D at various noise levels.

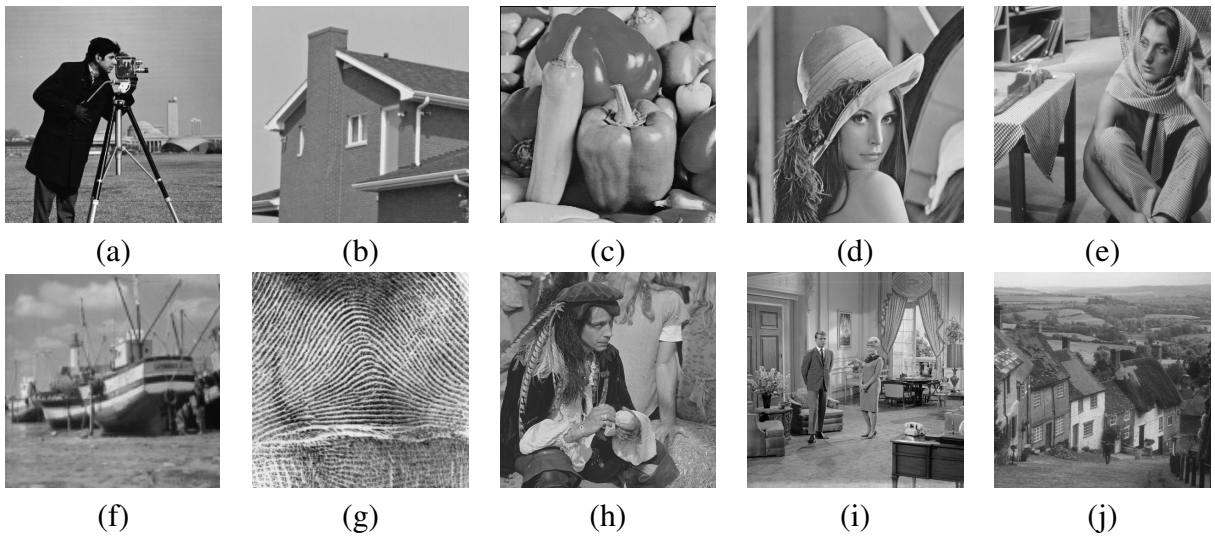


Figure 6.1: Single image dataset: (a) `Cameraman` image 256×256, (b) `House` image 256×256, (c) `Peppers` image 256×256, (d) `Lena` image 512×512, (e) `Barbara` image 512×512, (f) `Boats` image 512×512, (g) `Fingerprint` image 512×512, (h) `Man` image 512×512, (i) `Couple` image 512×512, and (j) `Hill` image 512×512.

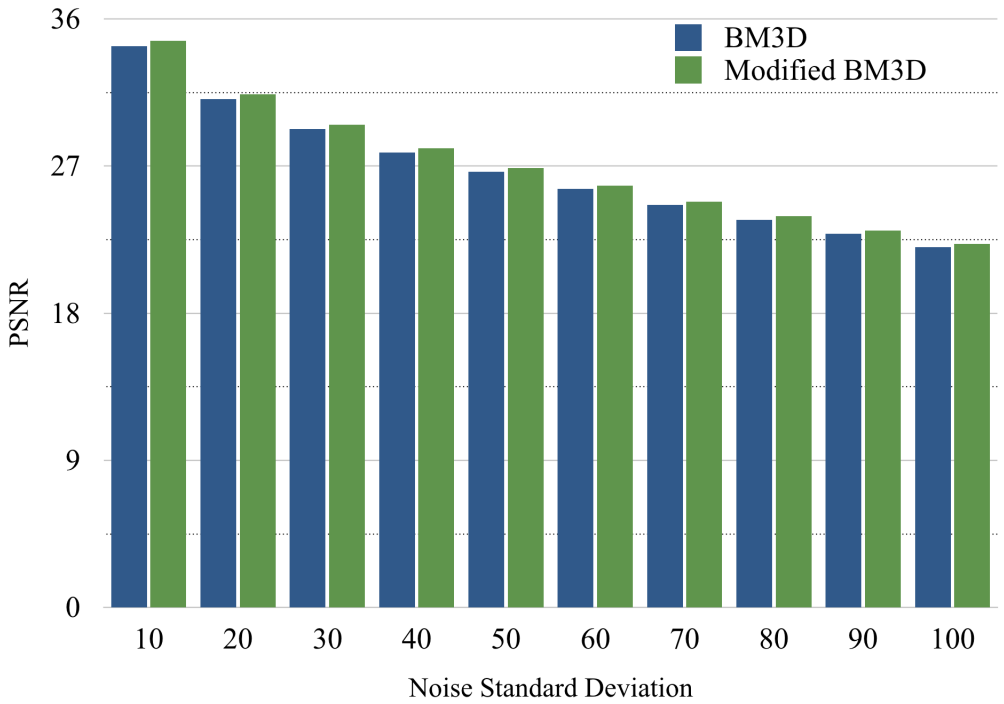


Figure 6.2: The PSNR performance of the single image denoising methods: BM3D, and modified BM3D when denoising the single images at varies noise levels (σ).

Table 6.1: The performance of patch-based single image denoising algorithms at various noise levels (σ). At each noise level (σ): the first row shows the results of using BM3D, and the second row shows the performance of our method.

σ	C.man	House	Pep-pers	Lena	Bar-bra	Boats	F.print	Man	Cou-ple	Hill	Aver-age
10	33.99	36.48	34.52	35.87	34.78	33.84	32.4	33.93	33.93	33.62	34.34
	34.38	**36.77**	**34.94**	**36.18**	**35.24**	**34.11**	**32.89**	**34.14**	**34.18**	**33.78**	**34.66**
20	30.21	33.6	31.14	33.01	31.59	30.77	28.82	30.59	30.63	30.68	31.10
	30.67	**33.89**	**31.52**	**33.30**	**31.95**	**31.03**	**29.09**	**30.78**	**30.90**	**30.91**	**31.40**
30	28.22	31.96	29.1	31.24	29.6	28.93	26.84	28.87	28.68	29.03	29.25
	28.58	**32.22**	**29.45**	**31.48**	**29.94**	**29.16**	**27.02**	**29.03**	**28.92**	**29.29**	**29.51**
40	26.38	30.57	27.42	29.93	28.14	27.54	25.52	27.6	27.24	27.86	27.82
	26.72	**30.86**	**27.73**	**30.18**	**28.47**	**27.73**	**25.64**	**27.80**	**27.47**	**28.09**	**28.07**
50	25.1	29.37	26.08	28.8	26.81	26.42	24.4	26.62	26.13	26.85	26.66
	25.38	**29.64**	**26.40**	**29.03**	**27.14**	**26.58**	**24.50**	**26.79**	**26.33**	**27.07**	**26.89**
60	23.86	28.15	24.88	27.71	25.58	25.46	23.44	25.73	25.18	25.95	25.59
	24.11	**28.50**	**25.25**	**27.95**	**25.92**	**25.57**	**23.52**	**25.88**	**25.35**	**26.11**	**25.81**
70	22.79	27.01	23.78	26.67	24.44	24.59	22.56	24.87	24.32	25.09	24.61
	23.03	**27.37**	**24.20**	**26.89**	**24.75**	**24.67**	**22.62**	**25.12**	**24.49**	**25.22**	**24.83**
80	21.87	25.94	22.74	25.66	23.36	23.79	21.74	24.05	23.56	24.27	23.70
	22.06	**26.27**	**23.20**	**25.87**	**23.68**	**23.92**	**21.82**	**24.29**	**23.72**	**24.40**	**23.92**
90	21.12	24.93	21.81	24.7	22.35	23.04	20.96	23.26	22.88	23.47	22.85
	21.20	**25.24**	**22.28**	**24.72**	**22.71**	**23.18**	**21.08**	**23.49**	**23.02**	**23.61**	**23.05**
100	20.29	23.92	20.99	23.78	21.45	22.36	20.23	22.52	22.26	22.72	22.05
	20.42	**24.24**	**21.37**	**23.80**	**21.81**	**22.49**	**20.38**	**22.74**	**22.37**	**22.86**	**22.25**

6.2.2 Qualitative Evaluation

The evaluation in this section is subjective, where the quality of the denoised images is addressed via the visual perception. Additive white Gaussian noisy images with the noise levels ($\sigma = 30, 90$) are chosen to perform this evaluation. Fragments of the denoised gray-scale `Man` and `House` images and the corresponding estimates are shown in Figure 6.3 and Figure 6.4, respectively.

The fragment images show that our method outperforms the BM3D method. Homogeneous regions are smoothed properly by our method; i.e., man's cheek in the `Man` image in Figure 6.3 (h). Our method preserves sharp edges; e.g., the door of the `House` image in Figure 6.4.

6.2.3 Discussion

Based on the results reported in Section 6.2, it is obvious that the proposed method outperforms the original BM3D method not only from the quantitative evaluation point of view, but also from the qualitative evaluation point of view. This is due to the adaptive hard thresholding and the addition of weights to the 3D array of the collaborative Wiener filtering.

The adaptive hard thresholding controls the level of thresholding based on a textures classification map. The weighted averaging adjusts the averaging of the 3D array of the collaborative Wiener filtering based on the patches' similarity. Therefore, more details of the true signal have been preserved, and more noise has been suppressed (i.e., preserved the textures and edges as well as smoothed the flat regions). On the other hand, the original BM3D hard thresholding, thresholds blindly the transforms coefficients where portions of valuable image details could be missing, and it retains the normal averaging for the 3D array of the collaborative Wiener filtering. Because BM3D is presently the state of the art denoising method, we believe that any improvement would contribute to developing a better denoising method. As this method has two denoising steps and more than twenty parameters, we improved just one parameter. However, improving all twenty parameters is needed in order to achieve a greater improvement.

6.3 New Multi-view Image Denoising Evaluation

This section provides an experimental study of the performance of the new patch-based methods for the multi-view image denoising methods. We used a fixed 5×5 patch size and a 9×9 search

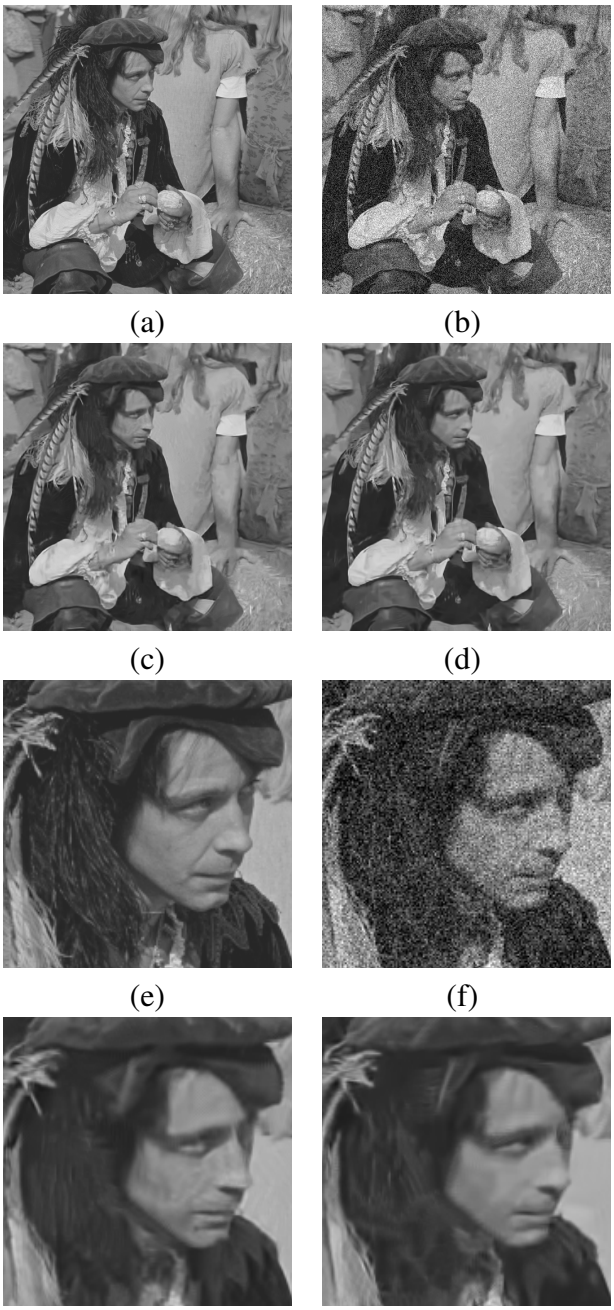


Figure 6.3: The results of denoising the `Man` image at noise level (σ = 30): (a) noise-free `Man` image 512×512, (b) AWGN noisy image (σ = 30), (c) BM3D, (d) our method, and (e), (f), (g) and (h) fragments images of the first rows

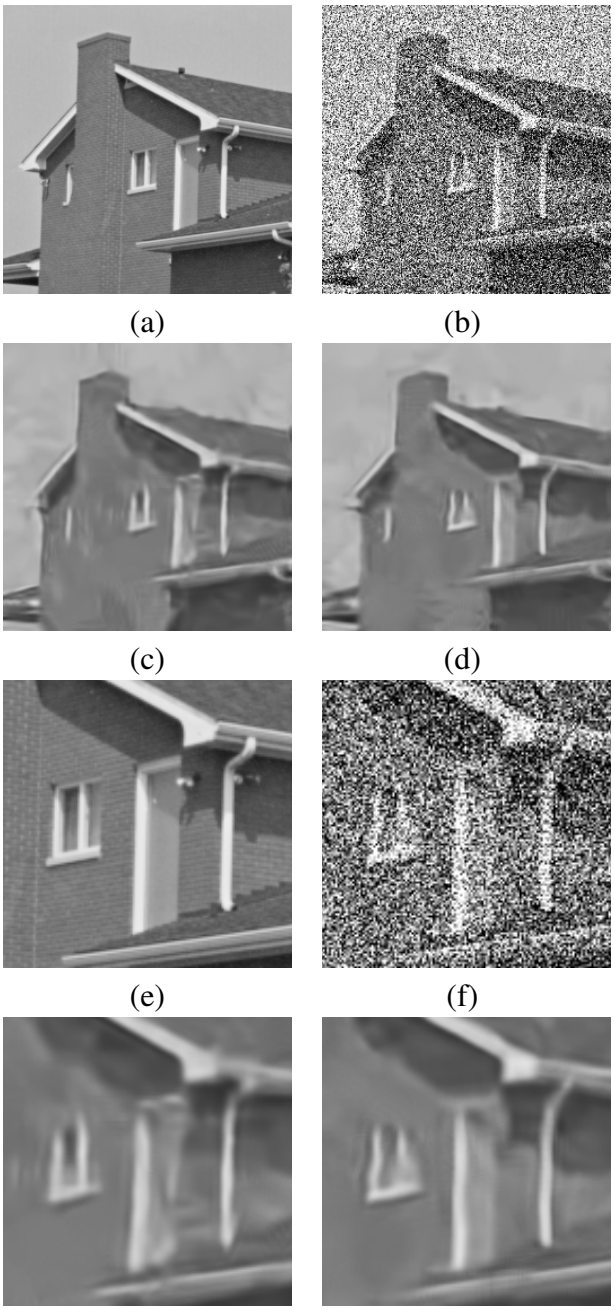


Figure 6.4: The results of denoising the `House` image at noise level (σ = 90): (a) noise-free `House` image 256×256, (b) AWGN noisy image (σ = 90), (c) BM3D, (d) our method, and (e), (f), (g) and (h) fragments images of the first rows

Figure 6.5: Multi-view image dataset: (a) `cones` images 450×375, (b) `teddy` images 450×375, (c) `tsukuba` images 384×288, and (d) `venus` images 434×383.

window size. Full searching was utilized for the S-SSIM method [7], and fixed bounded searching areas were employed with the size (-20, +20) for NL-Means with the MD similarity method (S-MD) and NL-Means with SVD-based similarity method (S-SVD) [8]. Four stereoscopic images were used in this experiment. The four images are grey-scale images, and are shown in Figure 6.5. In Subsection 6.3.1 and Subsection 6.3.2, the methods are evaluated both quantitatively and qualitatively, respectively.

6.3.1 Quantitative Evaluation

The experimental results of the proposed methods are shown in Table 6.2, which compares the performance of the proposed methods (S-SSIM, S-MD and S-SVD) to the original NL-Means method. A bold font with a wavy under-bar highlights the highest values of SSIM, while the highest values of PSNR are highlighted with a bold font.

Figure 6.6 shows the results of using the various methods on `Tsukuba` image at various noise levels. The chart shows that our method is the preferable one (from a MSSIM and PSNR point of view). Also the methods are preferable when they are used for denoising any of the four stereoscopic images at any noise level.

Table 6.2: The performance of patch-based multi-view image denoising algorithms at various noise levels (σ).

	σ	σ = 10		σ = 20		σ = 40		σ = 60	
	Method	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR
Cones	Noisy	0.572	25.06	0.353	21.13	0.165	15.91	0.097	12.91
	NL-Means	0.714	26.58	0.593	25.05	0.409	22.59	0.292	20.63
	S-SSIM	0.727	26.76	0.630	25.47	0.450	23.12	0.316	21.10
	S-MD	0.728	26.79	0.639	25.57	0.491	23.67	0.382	22.24
	S-SVD	0.729	**26.83**	0.640	**25.59**	0.496	**23.73**	0.387	**22.31**
Teddy	Noisy	0.543	25.68	0.317	21.42	0.147	16.16	0.087	13.19
	NL-Means	0.767	27.83	0.636	26.22	0.429	23.4	0.298	21.18
	S-SSIM	0.788	28.06	0.679	26.69	0.473	24.07	0.326	21.82
	S-MD	0.796	**28.12**	0.710	**26.92**	0.547	**24.78**	0.417	23.10
	S-SVD	0.795	28.11	0.709	26.89	0.546	**24.78**	0.419	**23.14**
Tsukuba	Noisy	0.588	25.63	0.367	21.67	0.183	16.79	0.106	13.82
	NL-Means	0.787	27.51	0.659	25.83	0.449	23.07	0.327	21.06
	S-SSIM	0.817	27.99	0.708	26.47	0.492	23.69	0.349	21.54
	S-MD	0.821	28.02	0.727	**26.58**	0.551	**24.26**	0.431	22.65
	S-SVD	0.821	**28.04**	0.724	26.53	0.552	24.25	0.431	**22.68**
Venus	Noisy	0.494	24.90	0.285	21.13	0.140	16.23	0.084	13.27
	NL-Means	0.748	26.74	0.614	25.44	0.399	22.88	0.280	20.94
	S-SSIM	0.767	26.93	0.650	25.81	0.436	23.56	0.300	21.54
	S-MD	0.780	26.98	0.694	25.99	0.519	24.09	0.394	22.58
	S-SVD	0.780	**26.99**	0.695	**26.00**	0.525	**24.14**	0.394	**22.59**

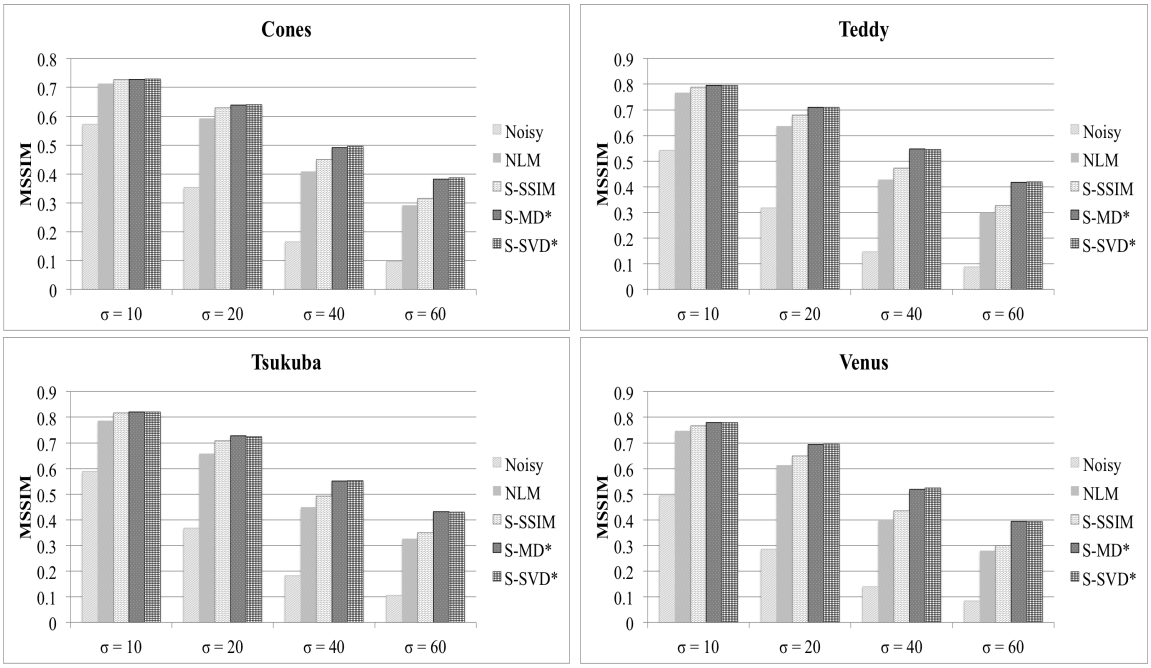


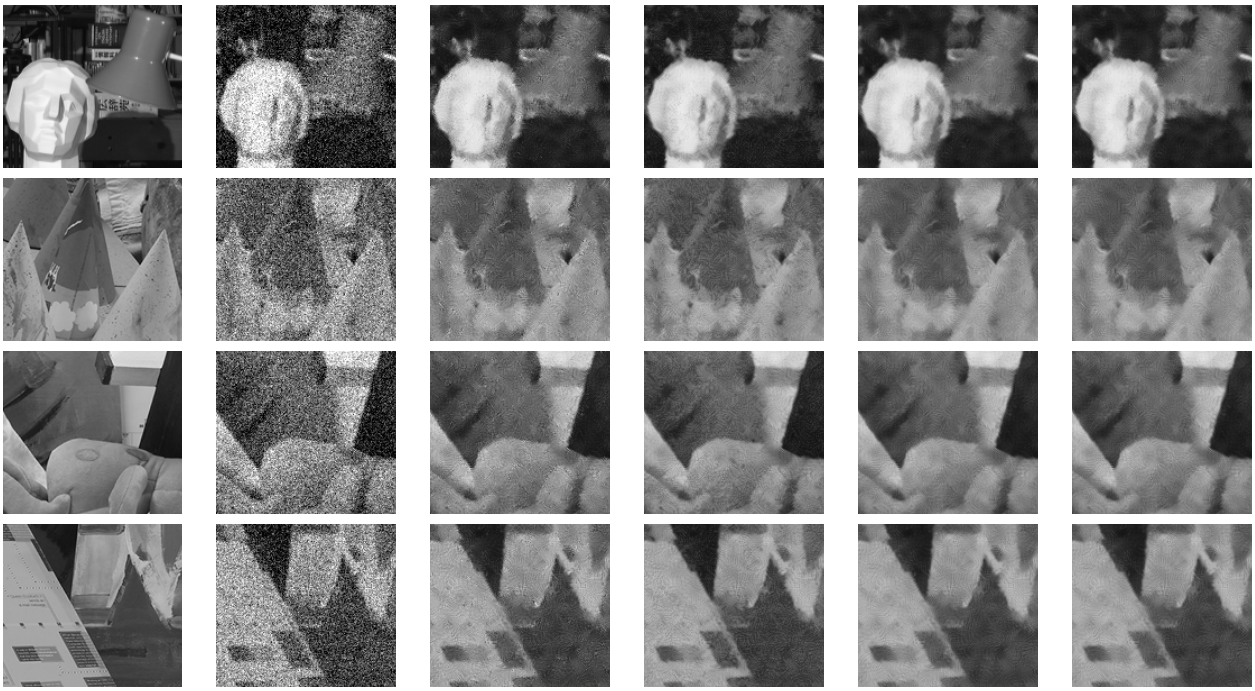
Figure 6.6: The MSSIM performance of the denoising methods: NL-Means, S-SSIM, S-MD, and S-SVD when denoising the four stereoscopic images at varies noise levels (σ).

6.3.2 Qualitative Evaluation

The methods are visually evaluated in this section. Here AWGN stereoscopic images with ($\sigma = 40$) were chosen in performing this evaluation. Fragments of the four noisy grayscale stereoscopic images and the corresponding estimates are shown in Figure 6.7. Each column shows the result of the same denoising method when applied to the dataset images. Figure 6.8 shows the disparity maps of these denoised images.

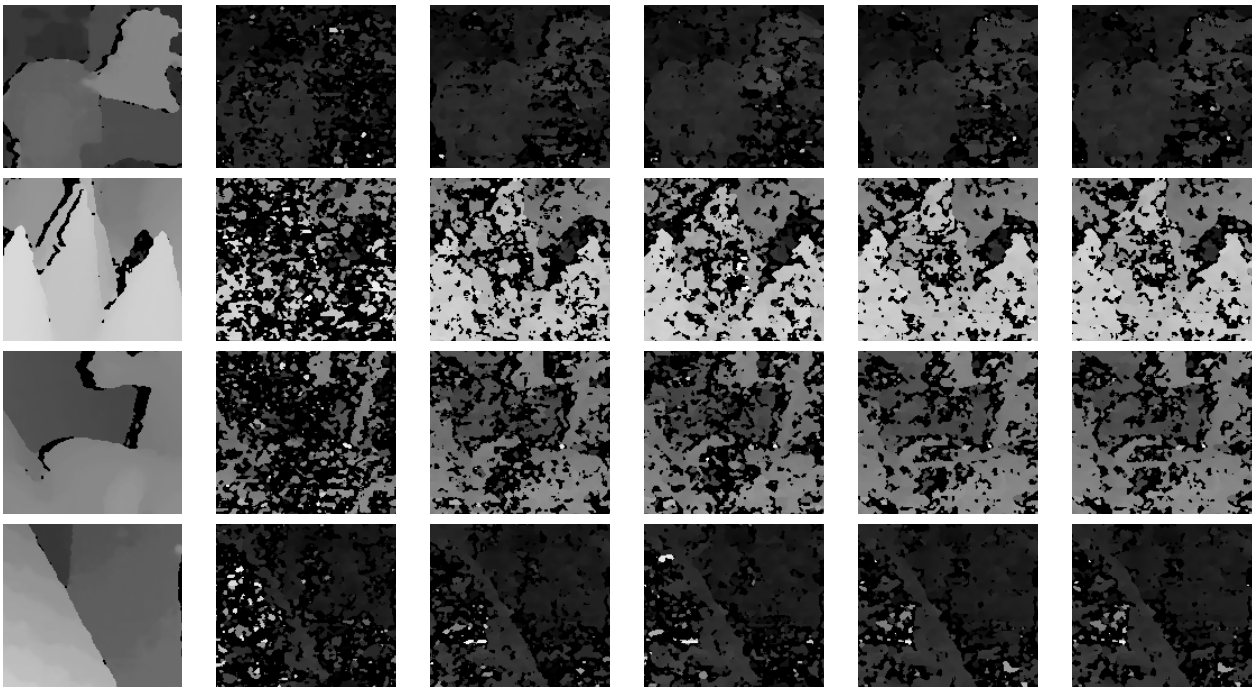
The fragment images appearing in Figure 6.7 show that our methods outperform other methods because our method preserves the sharp edges, e.g., the newspaper edges in `venus` image. Also the homogeneous regions are smoothed properly by our method, e.g., the head and lamp in `Tsukuba` image.

The fragments of the disparity maps in Figure 6.8 show that our method outperforms other methods. Our method produces disparity maps with fewer errors, e.g., the head and lamp in the disparity map of `Tsukuba` image. The full-sized images of this experiment with the accompanying disparity



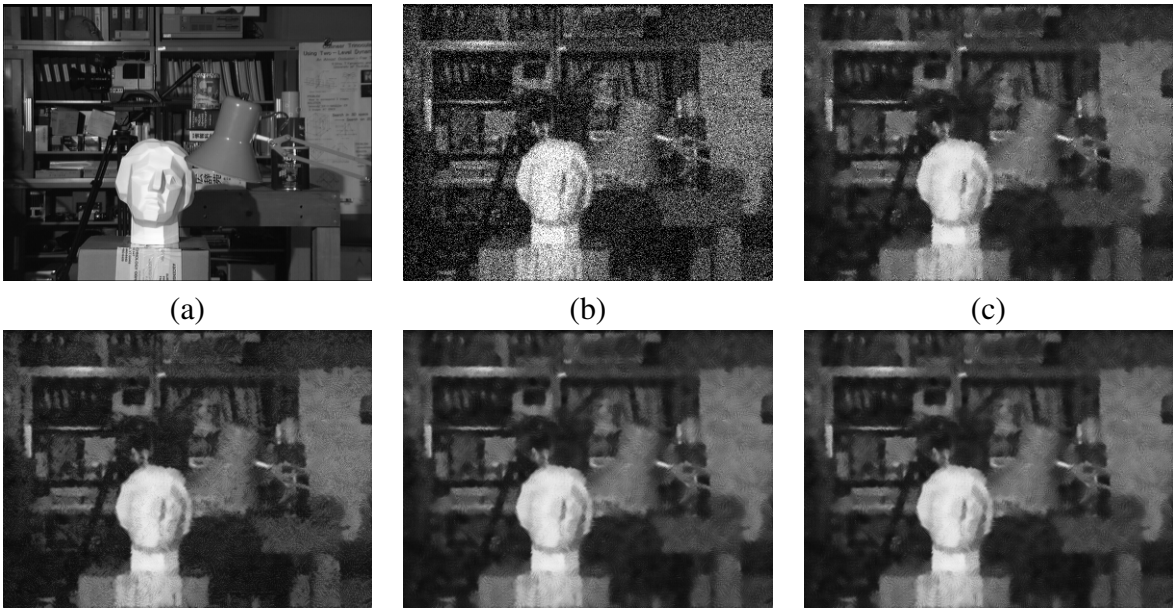
(a) (b) (c) (d) (e) (f)

Figure 6.7: Fragments of the noisy grayscale stereoscopic images: (a) Original images, (b) AWGN images (σ = 40), (c) NL-Means, (d) S-SSIM, (e) S-MD and (f) S-SVD



(a) (b) (c) (d) (e) (f)

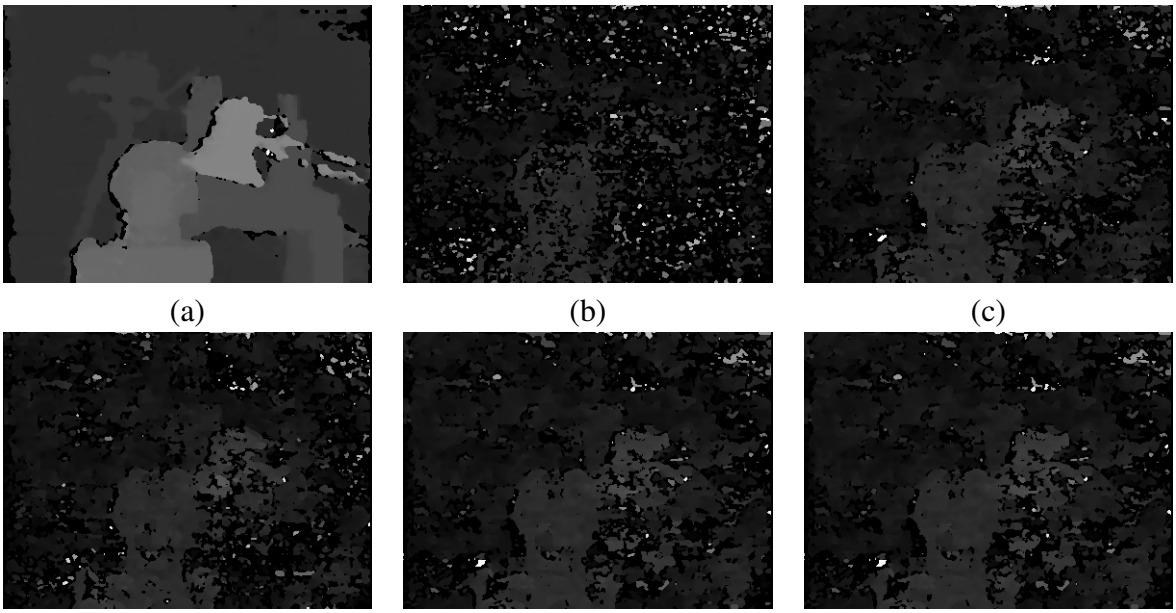
Figure 6.8: Fragments of the disparity map images: (a) Original images, (b) AWGN images (σ = 40), (c) NL-Means, (d) S-SSIM, (e) S-MD and (f) S-SVD



(a) (b) (c)

(d) (e) (f)

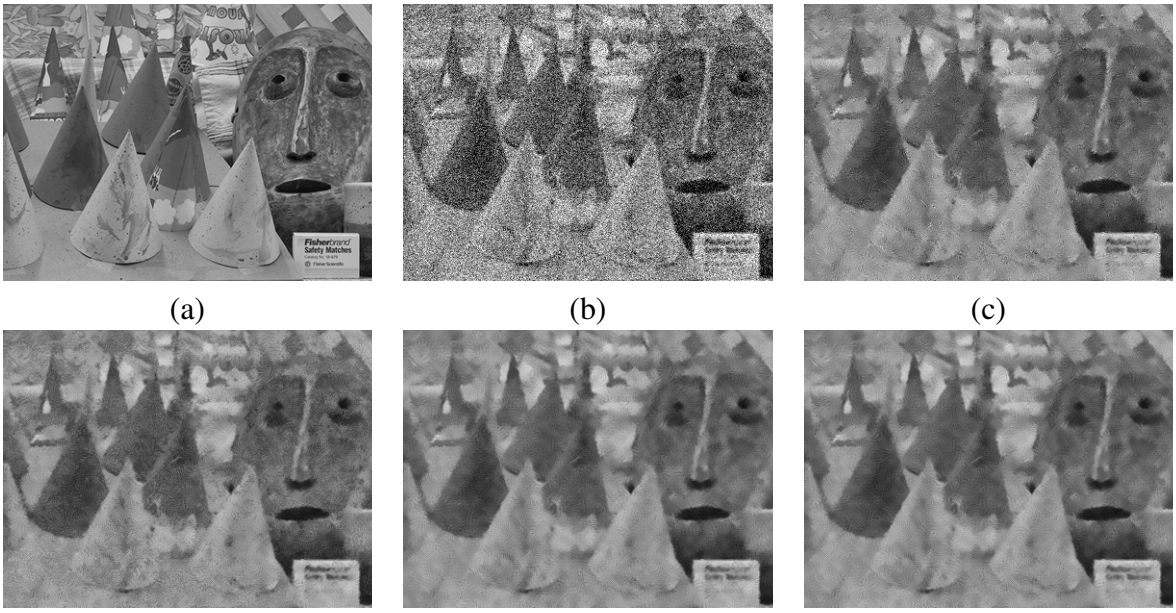
Figure 6.9: `Tsukuba` multi-view image denoising results: (a) Original images, (b) AWGN images (σ = 40), (c) NL-Means, (d) S-SSIM, (e) S-MD and (f) S-SVD



(a) (b) (c)

(d) (e) (f)

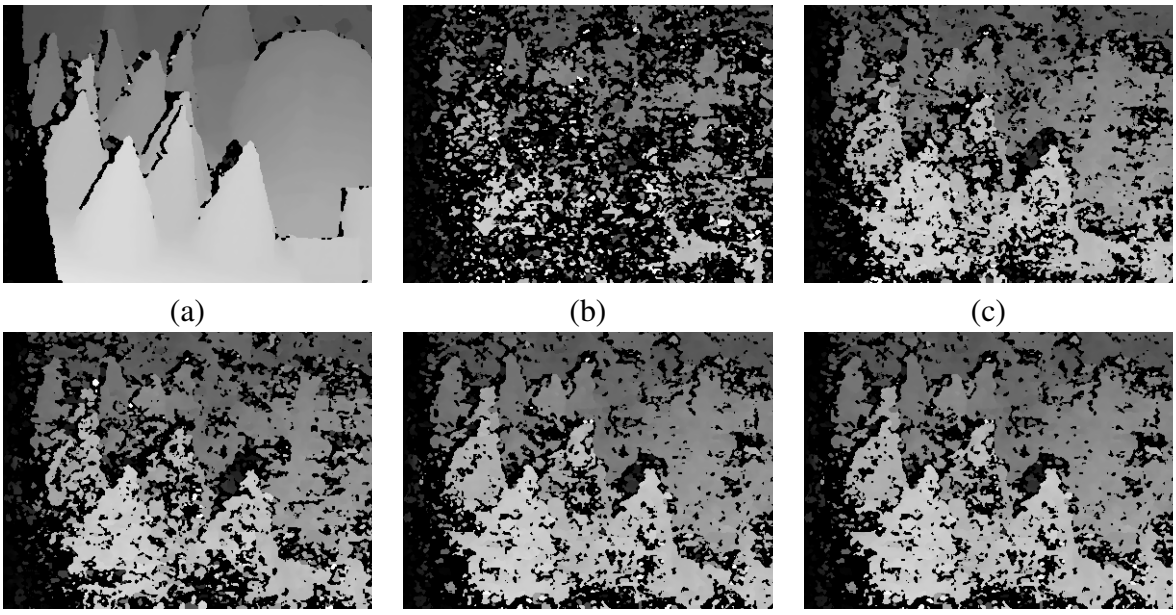
Figure 6.10: `Tsukuba` multi-view image denoising disparity maps: (a) Original image, (b) AWGN image (σ = 40), (c) NL-Means, (d) S-SSIM, (e) S-MD and (f) S-SVD



(a) (b) (c)

(d) (e) (f)

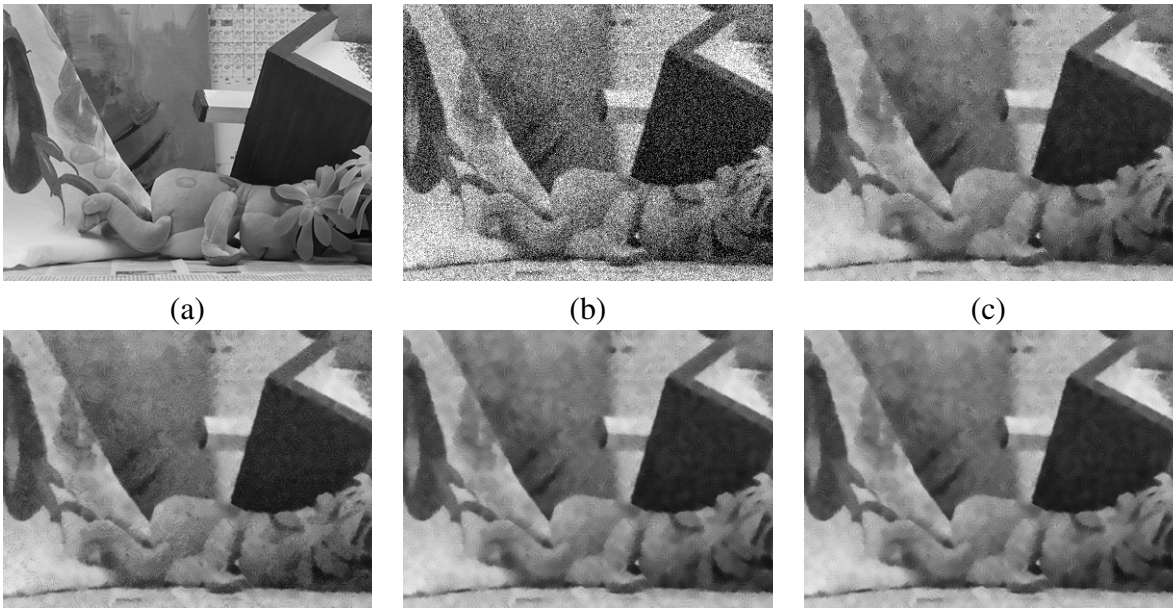
Figure 6.11: `Cones` multi-view image denoising results: (a) Original image, (b) AWGN image (σ = 40), (c) NL-Means, (d) S-SSIM, (e) S-MD and (f) S-SVD



(a) (b) (c)

(d) (e) (f)

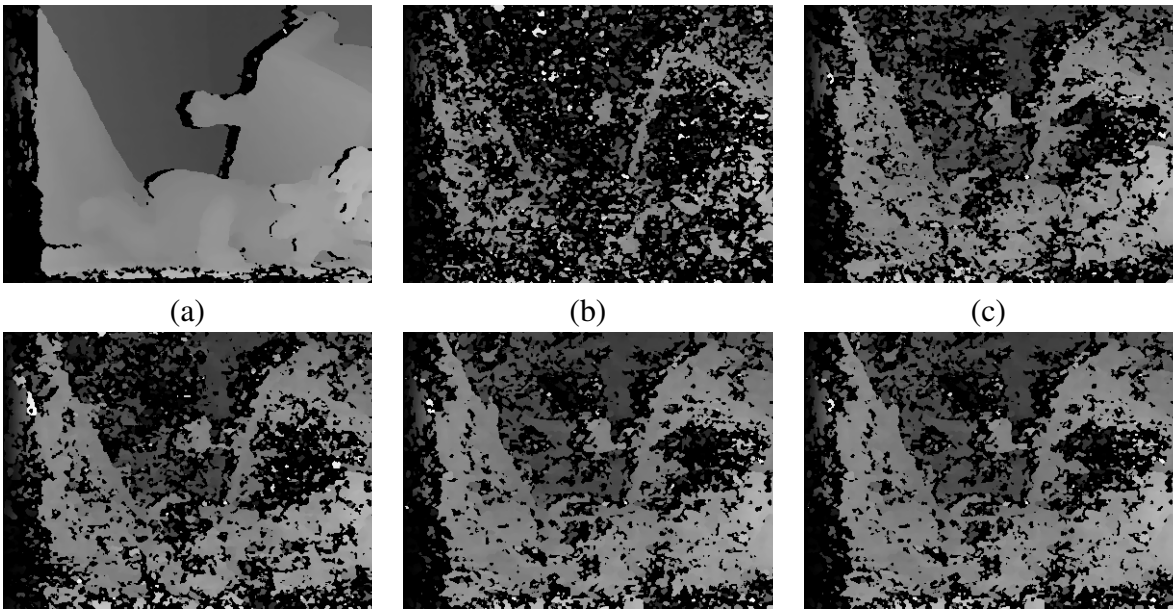
Figure 6.12: `Cones` multi-view image denoising disparity maps: (a) Original image, (b) AWGN image (σ = 40), (c) NL-Means, (d) S-SSIM, (e) S-MD and (f) S-SVD



(a) (b) (c)

(d) (e) (f)

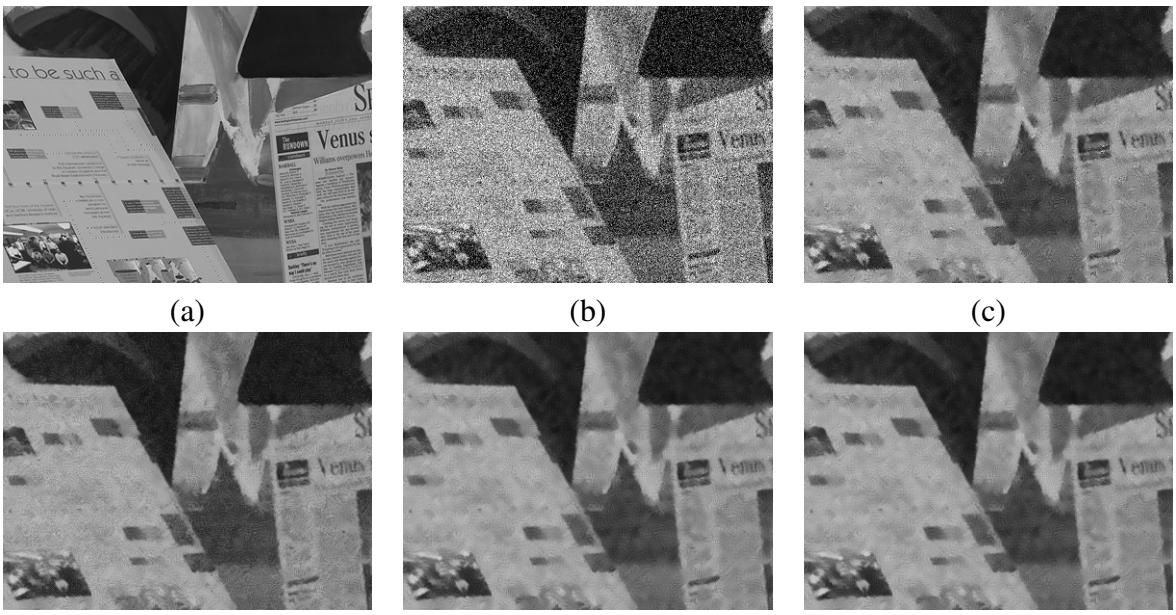
Figure 6.13: `Teddy` multi-view images denoising results: (a) Original image, (b) AWGN image (σ = 40), (c) NL-Means, (d) S-SSIM, (e) S-MD and (f) S-SVD



(a) (b) (c)

(d) (e) (f)

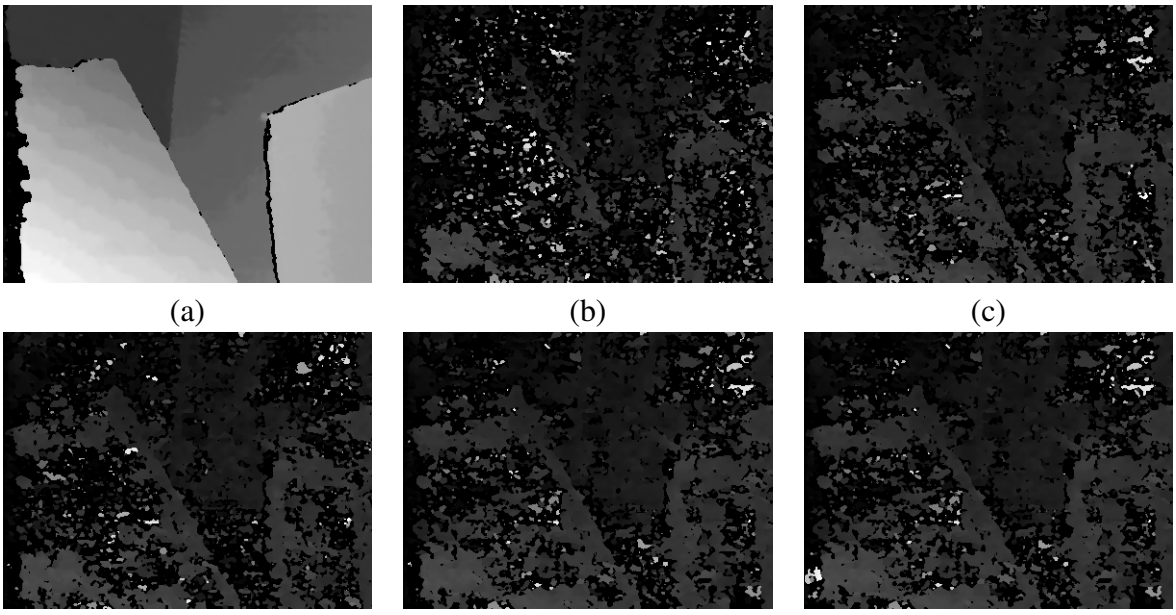
Figure 6.14: `Teddy` multi-view image denoising disparity maps: (a) Original image, (b) AWGN image (σ = 40), (c) NL-Means, (d) S-SSIM, (e) S-MD and (f) S-SVD



(a) (b) (c)

(d) (e) (f)

Figure 6.15: Venus multi-view image denoising results: (a) Original image, (b) AWGN image (σ = 40), (c) NL-Means, (d) S-SSIM, (e) S-MD and (f) S-SVD



(a) (b) (c)

(d) (e) (f)

Figure 6.16: Venus multi-view image denoising disparity maps: (a) Original image, (b) AWGN image (σ = 40), (c) NL-Means, (d) S-SSIM, (e) S-MD and (f) S-SVD

maps are shown in Figures 6.9, 6.10, 6.11, 6.12, 6.13, 6.14, 6.15, and 6.16.

6.3.3 Discussion

The results discussed in this section show that image similarity measures assist greatly in the performance of multi-view image denoising methods. As was previously mentioned in the review chapter of this thesis, the quality of the image similarity measures depends on the image processing applications used.

The SSIM index is one of the best image quality assessment techniques; however, it unfortunately fails when it is used with noisy images. Indeed, the *structural similarity* index would favour comparing similar noise patterns rather than the actual structures of the images. Here S-SVD overcomes the issue of S-SSIM by using a transformation before measuring the three changes: the luminance, contrast, and structure. Thus, the results obtained when using the S-MD and the S-SVD methods are better than those obtained when using the S-SSIM denoising method.

One of the problems associated with multi-view image denoising methods is the long execution time due to the large number of images to be searched when filtering. Hence, the use of a bounded searching area is recommended to overcome this disadvantage. In contrast to the S-SSIM filtering method, the S-MD and the S-SVD filtering methods employ bounded searching areas.

Chapter 7

Conclusions and Future Work

In this thesis, we started by reviewing the background of the problem in order to establish an understanding of the context of the denoising problem. Then, a statement of the denoising problem was provided. Next, the purpose of the research and the motivation behind it was explained. Before presenting the new patch-based denoising method, a comprehensive review of patch-based denoising methods was introduced. After that, the experimental results achieved were presented, as well as a comparison with other denoising methods. This last chapter will conclude this work based on the experimental observations made during this study, and present recommendations for further improvements.

7.1 Summary of Contributions

7.1.1 Single Image Denoising

In this work, a new patch-based denoising method for single images was presented. Improving the thresholding step and adjusting the accuracy of the aggregation stage for the collaborative Wiener filtering step improved the performance of the state-of-the-art single image denoising method (BM3D filtering method). Based on a comprehensive quantitative and qualitative evaluation of the new methods and of the existing ones, we conclude that:

1. A *modified* BM3D filtering algorithm for denoising noisy images was developed.
2. The *modified* BM3D filtering algorithm outperforms the original BM3D filtering algorithm

at various noise levels.

3. Edges were preserved and the homogeneous regions were properly smoothed using the *modified* BM3D filtering algorithm.

4. The idea that the pixel-based BF could be adapted successfully to patch-based denoising methods was confirmed.

5. Utilizing geometric and luminance distances between similar patches assists in producing an adaptive hard-thresholding operator.

6. The texture analysis approach (GLCM) can be adapted locally for patches; in addition, it can be utilized for improving original BM3D filtering algorithm.

7. Using adaptive hard thresholding reduces the number of parameters of the original BM3D by more than twenty parameters.

8. Weighted averaging improves the aggregation stage of the collaborative Wiener filtering step.

9. The proposed method performs better with some wavelet transforms (i.e., `bior1.1`, `bior1.3`, `rbio1.1`, and `db1`).

7.1.2 Multi-views Image Denoising

This thesis also presented new patch-based denoising method for reducing noise in multi-view images. Image similarity measures significantly affect the performance of multi-view image denoising methods. SSIM, MD and SVD-based image similarity measures outperform other denoising methods at various levels of noise (σ) . Based on extensive experimental results, the following observations can be made:

1. The NL-Means can be adopted with various image similarity measures to denoise multi-view images.

2. The SSIM index with the NL-Means for stereoscopic image denoising produced acceptable results at low levels of noise ($\sigma \leqslant 20$), but it failed to properly denoise the image at a high level of noise ($\sigma > 20$).

3. The SSIM index compares noise structure instead of patch structures.

4. The transform used in SVD-based image similarity measure assists in producing a robust image similarity measure.

5. A bounded search area assists in speeding up the filtering of multi-view images.

6. Using NL-Means with MD and SVD-based image similarity denoises both left and right images simultaneously and in a shorter time.

7.2 Future Work

The patch-based denoising approaches introduced in this thesis are modest, efficient and reliable. However, these approaches might need some further work in order to improve the quality of their overall output.

Because BM3D is the state of the art denoising method, we believe that any improvement to it would contribute to developing a better denoising method. It has two denoising steps and more than twenty parameters; we improved just one parameter in the *modified* BM3D filtering algorithm. Improving the twenty parameters is necessary in order to achieve an even greater improvement. While we improved the performance of BM3D, we believe that more effort is needed in order to improve the speed of this method. BM3D can be parallelized in order to improve its efficiency.

Our proposed method for multi-view image denoising is based on the use of spatial domain filters. We believe that using a transform domain for the filtering of such images would improve their performance. The methods use traditional image similarity measures (e.g. MSE) for assigning the weights between similar patches. Recently, some new similarity measures have been developed that compete with the traditional MSE. We believe that using the more modern similarity measures for assigning weights would help to improve the results of the methods used.

Bibliography

[1] (1977) The university of southern california - signal and image processing institute (usc-sipi) image database. [Online]. Available: http://sipi.usc.edu/database/

[2] (2008) Images from digital image processing, 3rd ed, by gonzalez and woods. [Online]. Available: http://www.imageprocessingplace.com/root_files_V3/image_databases.htm

[3] (2015, June) Middlebury stereo datasets - middlebury computer vision. [Online]. Available: http://vision.middlebury.edu/stereo/data/

[4] S. Aghagolzadeh and O. K. Ersoy, "Transform image enhancement," *The International Society for Optical Engineering*, vol. 31, no. 3, pp. 614–626, 1992. [Online]. Available: http://dx.doi.org/10.1117/12.56095

[5] M. Aharon, M. Elad, and A. Bruckstein, "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *Signal Processing, IEEE Transactions on*, vol. 54, no. 11, pp. 4311–4322, 2006.

[6] S. Akramullah, *Digital Video Concepts, Methods, and Metrics Quality, Compression, Performance, and Power Trade-off Analysis*. Berkeley, CA: Apress, 2014, ch. Digital Video Compression Techniques, pp. 11–54. [Online]. Available: http://dx.doi.org/10.1007/978-1-4302-6713-3_8

[7] M. H. Alkinani and M. R. El-Sakka, "Non-local means for stereo image denoising using structural similarity," in *Image Analysis and Recognition: 12th International Conference, ICIAR 2015*, ser. Lecture Notes in Computer Science, M. Kamel and A. Campilho, Eds. Springer International Publishing, 2015, vol. 9164, pp. 51–59. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-20801-5_6

[8] ——, "Denoising multi-view images using non-local means with different similarity measures," in *Image Analysis and Recognition: 13th International Conference, ICIAR 2016, in Memory of Mohamed Kamel, Póvoa de Varzim, Portugal, July 13-15, 2016,*

Proceedings, A. Campilho and F. Karray, Eds. Cham: Springer International Publishing, 2016, pp. 101–109. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-41501-7_12

[9] ——, "A modified block matching 3d algorithm for additive noise reduction," *Mathematics for Applications*, vol. 5, no. 2, pp. 93–103, 2016. [Online]. Available: http://ma.fme.vutbr.cz/archiv/5_2/ma_5_2_alkinani_el_sakka_final.pdf

[10] A. Atto, D. Pastor, and G. Mercier, "Wavelet shrinkage: unification of basic thresholding functions and thresholds," vol. 5, no. 1, pp. 11–28, 2011. [Online]. Available: http://dx.doi.org/10.1007/s11760-009-0139-y

[11] S. Bacchelli and S. Papi, "Image denoising using principal component analysis in the wavelet domain," *Journal of Computational and Applied Mathematics*, vol. 189, no. 1–2, pp. 606 – 621, 2006, proceedings of The 11th International Congress on Computational and Applied Mathematics.

[12] F. Bashar and M. R. El-Sakka, "Bm3d image denoising using learning-based adaptive hard thresholding," in *International Conference on Computer Vision Theory and Applica- tions (VISAPP)*, 2016.

[13] E. P. Bennett and L. McMillan, "Video enhancement using per-pixel virtual exposures," in *ACM SIGGRAPH 2005 Papers*, ser. SIGGRAPH '05. New York, NY, USA: ACM, 2005, pp. 845–852. [Online]. Available: http://doi.acm.org/10.1145/1186822.1073272

[14] G. Beylkin, R. Coifman, and V. Rokhlin, "Fast wavelet transforms and numerical algorithms i," *Communications on Pure and Applied Mathematics*, vol. 44, no. 2, pp. 141–183, 1991. [Online]. Available: http://dx.doi.org/10.1002/cpa.3160440202

[15] J. Boussinesq, *Théorie analytique de la chaleur*. Paris: Gauthier-Villars, 1903.

[16] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 9, pp. 1124–1137, 2004.

[17] A. Buades, B. Coll, and J. M. Morel, "A non-local algorithm for image denoising," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 2, 2005, pp. 60–65 vol. 2.

[18] C. Chan, R. Fulton, D. D. Feng, and S. Meikle, "Median non-local means filtering for low snr image denoising: Application to pet with anatomical knowledge," in *IEEE Nuclear Science Symposuim Medical Imaging Conference*, Oct 2010, pp. 3613–3618.

[19] S. H. Chan, D. T. Võ, and T. Q. Nguyen, "Subpixel motion estimation without interpolation," in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, March 2010, pp. 722–725.

[20] S. Chang, B. Yu, and M. Vetterli, "Adaptive wavelet thresholding for image denoising and compression," *Image Processing, IEEE Transactions on*, vol. 9, no. 9, pp. 1532–1546, 2000.

[21] ——, "Spatially adaptive wavelet thresholding with context modeling for image denoising," *Image Processing, IEEE Transactions on*, vol. 9, no. 9, pp. 1522–1531, 2000.

[22] J. S. Charles-Alban Deledalle and A. Dalalyan, "Image denoising with patch based pca: local versus global," in *Proceedings of the British Machine Vision Conference*, J. Hoey, S. McKenna, and E. Trucco, Eds. BMVA Press, 2011, pp. 25.1–25.10.

[23] Y. C. Chen, V. M. Patel, P. J. Phillips, and R. Chellappa, "Dictionary-based face and person recognition from unconstrained video," *IEEE Access*, vol. 3, pp. 1783–1798, 2015.

[24] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *Image Processing, IEEE Transactions on*, vol. 16, no. 8, pp. 2080–2095, 2007.

[25] ——, "Joint image sharpening and denoising by 3d transform-domain collaborative filtering," in *Proc. 2007 int. Ticsp workshop spectral meth. Multirate signal process., smmsp 2007*, 2007.

[26] ——, "Bm3d image denoising with shape-adaptive principal component analysis," in *Proc. Workshop On Signal Processing With Adaptive Sparse Structured Representations (SPARS'09)*, 2009.

[27] A. Danielyan, A. Foi, V. Katkovnik, and K. Egiazarian, "Image and video super-resolution via spatially adaptive blockmatching filtering," in *Proceedings of International Workshop on Local and Non-Local Approximation in Image Processing (LNLA)*, 2008.

[28] ——, "Image upsampling via spatially adaptive block-matching filtering," in *Proc. Of 16th European Signal Processing Conference, Eusipco2008*, 2008.

[29] I. Daubechies, *Ten Lectures on Wavelets*. Society for Industrial and Applied Mathematics, 1992. [Online]. Available: http://epubs.siam.org/doi/abs/10.1137/1.9781611970104

[30] ——, "Orthonormal bases of compactly supported wavelets ii. variations on a theme," *SIAM Journal on Mathematical Analysis*, vol. 24, no. 2, pp. 499–519, 1993. [Online]. Available: http://dx.doi.org/10.1137/0524031

[31] I. Daubechies, A. Grossmann, and Y. Meyer, "Painless nonorthogonal expansions," *Journal of Mathematical Physics*, vol. 27, no. 5, pp. 1271–1283, 1986. [Online]. Available: http://scitation.aip.org/content/aip/journal/jmp/27/5/10.1063/1.527388

[32] C.-A. Deledalle, L. Denis, and F. Tupin, "Iterative weighted maximum likelihood denoising with probabilistic patch-based weights," *Image Processing, IEEE Transactions on*, vol. 18, no. 12, pp. 2661–2672, 2009.

[33] D. L. Donoho and I. M. Johnstone, "Ideal spatial adaptation by wavelet shrinkage," *Biometrika*, vol. 81, pp. 425–455, 1994.

[34] ——, "Adapting to unknown smoothness via wavelet shrinkage," *Journal of the american statistical association*, vol. 90, no. 432, pp. 1200–1224, 1995.

[35] D. L. Donoho and J. M. Johnstone, "Ideal spatial adaptation by wavelet shrinkage," *Biometrika*, vol. 81, no. 3, pp. 425–455, 1994.

[36] D. Donoho, "De-noising by soft-thresholding," *Information Theory, IEEE Transactions on*, vol. 41, no. 3, pp. 613–627, 1995.

[37] F. Durand and J. Dorsey, "Fast bilateral filtering for the display of high-dynamic-range images," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 257–266, Jul. 2002. [Online]. Available: http://doi.acm.org/10.1145/566654.566574

[38] A. E. P. Enríquez and V. Ponomaryov, "Image denoising using block matching and discrete cosine transform with edge restoring," in *2016 International Conference on Electronics, Communications and Computers (CONIELECOMP)*, Feb 2016, pp. 140–147.

[39] X. Fei, W. Huang, K. Wang, and Z. Wei, "Patch-based image denoising with geometric structure clustering," in *Advances in Image and Graphics Technologies*, ser. Communications in Computer and Information Science, T. Tan, Q. Ruan, X. Chen, H. Ma, and L. Wang, Eds. Springer Berlin Heidelberg, 2013, vol. 363, pp. 85–91.

[40] A. Foi, V. Katkovnik, and K. Egiazarian, "Pointwise shape-adaptive dct for high-quality denoising and deblocking of grayscale and color images," *Image Processing, IEEE Transactions on*, vol. 16, no. 5, pp. 1395–1411, May 2007.

[41] Y. Fu, A. Lam, I. Sato, and Y. Sato, "Adaptive spatial-spectral dictionary learning for hyperspectral image denoising," in *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec 2015, pp. 343–351.

[42] L. Genin, F. Champagnat, G. L. Besnerais, and L. Coret, "Point object detection using a nl-means type filter," in *2011 18th IEEE International Conference on Image Processing*, Sept 2011, pp. 3533–3536.

[43] A. Gersho, "On the structure of vector quantizers," *Information Theory, IEEE Transactions on*, vol. 28, no. 2, pp. 157–166, 1982.

[44] G. Gomez, "Estimating local variance for gaussian filtering," *Contribution to CV-Online: On-Line Compendium of Computer Vision, R. Fisher, ed*, 2002.

[45] A. Haar, "Zur Theorie der orthogonalen Funktionensysteme," *Mathematische Annalen*, vol. 69, no. 3, pp. 331–371, Sep. 1910. [Online]. Available: http://dx.doi.org/10.1007/bf01456326

[46] K. M. Hanson, "Simplified method of estimating noise-power spectra," in *Medical Imaging'98*. International Society for Optics and Photonics, 1998, pp. 243–250.

[47] M. Hasan and M. R. El-Sakka, *Structural Similarity Optimized Wiener Filter: A Way to Fight Image Noise*. Cham: Springer International Publishing, 2015, pp. 60–68. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-20801-5_7

[48] R. Hedjam, R. F. Moghaddam, and M. Cheriet, "Markovian clustering for the non-local means image denoising," in *2009 16th IEEE International Conference on Image Processing (ICIP)*, Nov 2009, pp. 3877–3880.

[49] Y. S. Heo, K. M. Lee, and S. U. Lee, "Simultaneous depth reconstruction and restoration of noisy stereo images using non-local pixel distribution," in *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, June 2007, pp. 1–8.

[50] A. Hore and D. Ziou, "Image quality metrics: Psnr vs. ssim," in *Pattern Recognition (ICPR), 2010 20th International Conference on*, 2010, pp. 2366–2369.

[51] Y. Hou, C. Zhao, D. Yang, and Y. Cheng, "Comments on "image denoising by sparse 3-d transform-domain collaborative filtering," *Image Processing, IEEE Transactions on*, vol. 20, no. 1, pp. 268–270, Jan 2011.

[52] D. A. Huang, L. W. Kang, Y. C. F. Wang, and C. W. Lin, "Self-learning based image decomposition with applications to single image denoising," *IEEE Transactions on Multimedia*, vol. 16, no. 1, pp. 83–93, Jan 2014.

[53] J. Immerkaer, "Fast noise variance estimation," *Computer vision and image understanding*, vol. 64, no. 2, pp. 300–302, 1996.

[54] P. Irrera, I. Bloch, and M. Delplanque, "A flexible patch based approach for combined denoising and contrast enhancement of digital x-ray images," *Medical Image Analysis*, vol. 28, pp. 33–45, 2 2016. [Online]. Available: //www.sciencedirect.com/science/article/pii/S1361841515001681

[55] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, Sep. 1999.

[56] W. James and C. Stein, "Estimation with quadratic loss," in *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*. Berkeley, Calif.: University of California Press, 1961, pp. 361–379. [Online]. Available: http://projecteuclid.org/euclid.bsmsp/1200512173

[57] F. Jin, P. Fieguth, L. Winger, and E. Jernigan, "Adaptive wiener filtering of noisy images and image sequences," in *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, vol. 3. IEEE, 2003, pp. III–349.

[58] L. W. Kang, C. Y. Hsu, H. W. Chen, C. S. Lu, C. Y. Lin, and S. C. Pei, "Feature-based sparse representation for image similarity assessment," *IEEE Transactions on Multimedia*, vol. 13, no. 5, pp. 1019–1030, Oct 2011.

[59] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1988. [Online]. Available: http://dx.doi.org/10.1007/BF00133570

[60] A. Khan and M. R. El-Sakka, "Non-local means using adaptive weight thresholding," in *Proceedings of the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, 2016, pp. 67–76.

[61] M. Kim, D. Park, D. K. Han, and H. Ko, "A novel approach for denoising and enhancement of extremely low-light video," *IEEE Transactions on Consumer Electronics*, vol. 61, no. 1, pp. 72–80, February 2015.

[62] T. Kohonen, M. R. Schroeder, and T. S. Huang, *Self-Organizing Maps*, 3rd ed. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2001.

[63] R. Lai and X. x. Dou, "Improved non-local means filtering algorithm for image denoising," in *Image and Signal Processing (CISP), 2010 3rd International Congress on*, vol. 2, Oct 2010, pp. 720–722.

[64] M. Lebrun, "An Analysis and Implementation of the BM3D Image Denoising Method," *Image Processing On Line*, vol. 2012, pp. 175–213, 2012.

[65] J.-S. Lee, "Digital image smoothing and the sigma filter," *Computer Vision, Graphics, and Image Processing*, vol. 24, no. 2, pp. 255 – 269, 1983. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0734189X83900476

[66] X. Liu, M. Tanaka, and M. Okutomi, "Noise level estimation using weak textured patches of a single noisy image," in *2012 19th IEEE International Conference on Image Processing*. IEEE, 2012, pp. 665–668.

[67] ——, "Single-image noise level estimation for blind denoising," *IEEE transactions on image processing*, vol. 22, no. 12, pp. 5226–5237, 2013.

[68] Y.-S. Liu, P.-Q. Yu, J.-H. Yong, H. Zhang, and J.-G. Sun, "Bilateral filter for meshes using new predictor," in *Proceedings of the First international conference on Computational and Information Science*, ser. CIS'04. Berlin, Heidelberg: Springer-Verlag, 2004, pp. 1093–1099. [Online]. Available: $http://dx.doi.org/10.1007/978-3-540-30497-5_168$

[69] E. Luo, S. H. Chan, S. Pan, and T. Q. Nguyen, "Adaptive non-local means for multiview image denoising: Searching for the right patches via a statistical approach," in *2013 IEEE International Conference on Image Processing*, Sept 2013, pp. 543–547.

[70] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. Fifth Berkeley Symp. on Math. Statist. and Prob.*, vol. 1. Univ. of Calif. Press, 1967, pp. 281–297.

[71] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Non-local sparse models for image restoration," in *Computer Vision, 2009 IEEE 12th International Conference on*, 2009, pp. 2272–2279.

[72] J. Mairal, G. Sapiro, and M. Elad, "Learning multiscale sparse representations for image and video restoration," Tech. Rep., 2007.

[73] S. G. Mallat, "Multiresolution approximations and wavelet orthonormal bases of l²(r)," *Transactions of the American Mathematical Society*, vol. 315, no. 1, pp. 69–87, 1989. [Online]. Available: http://www.jstor.org/stable/2001373

[74] C. D. Manning, P. Raghavan, and H. Schutze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008.

[75] G. M. Maruf and M. R. El-Sakka, *Improved Non-Local Means Algorithm Based on Dimensionality Reduction*. Cham: Springer International Publishing, 2015, pp. 43–50. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-20801-5_5

[76] H. B. Mitchell, *Image Similarity Measures*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 167–185. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-11216-4_14

[77] D. Muresan and T. Parks, "Adaptive principal components and image denoising," in *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, vol. 1, 2003, pp. I–101–4 vol.1.

[78] D. Muti, S. Bourennane, and M. Guillaume, "Svd-based image filtering improvement by means of image rotation," in *Acoustics, Speech, and Signal Processing, 2004. Proceedings. (ICASSP '04). IEEE International Conference on*, vol. 3, 2004, pp. iii–289–92 vol.3.

[79] M. Narwaria and W. Lin, "Svd-based quality metric for image and video using machine learning," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 42, no. 2, pp. 347–364, April 2012.

[80] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, no. 6583, pp. 607–609, Jun. 1996.

[81] S. Pal, R. Mahakud, and M. Sahoo, "Pca based image denoising using lpg," *IJCA Special Issue on 2nd National Conference- Computing, Communication and Sensor Network (CCSN)*, no. 3, pp. 20–25, 2011, published by Foundation of Computer Science, New York, USA.

[82] V. Papyan and M. Elad, "Multi-scale patch-based image restoration," *IEEE Transactions on Image Processing*, vol. 25, no. 1, pp. 249–261, Jan 2016.

[83] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 12, no. 7, pp. 629–639, Jul 1990.

[84] G. Peyré and J. Fadili, "Learning analysis sparsity priors," in *Proc. of Sampta'11*, 2011. [Online]. Available: http://hal.archives-ouvertes.fr/hal-00542016/

[85] T. Pham and L. van Vliet, "Separable bilateral filtering for fast video preprocessing," in *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, 2005, pp. 4 pp.–.

[86] M. Poderico, S. Parrilli, G. Poggi, and L. Verdoliva, "Sigmoid shrinkage for bm3d denoising algorithm," in *Multimedia Signal Processing (MMSP), 2010 IEEE International Workshop on*, Oct 2010, pp. 423–426.

[87] J. Polzehl and V. Spokoiny, "Propagation-separation approach for local likelihood estimation," *Probability Theory and Related Fields*, vol. 135, no. 3, pp. 335–362, 2006.

[88] J. Polzehl and K. Tabelow, "Adaptive smoothing of digital images: The r package adimpro," *Journal of Statistical Software*, vol. 19, no. i01, undated.

[89] O. Rioul and M. Vetterli, "Wavelets and signal processing," *Signal Processing Magazine, IEEE*, vol. 8, no. 4, pp. 14–38, 1991.

[90] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1–4, pp. 259 – 268, 1992. [Online]. Available: http://www.sciencedirect.com/science/article/pii/016727899290242F

[91] Z. Shi, B. Xu, X. Zheng, and M. Zhao, "A chinese character structure preserved denoising method for chinese tablet calligraphy document images based on ksvd dictionary learning," *Multimedia Tools and Applications*, pp. 1–16, 2016. [Online]. Available: http://dx.doi.org/10.1007/s11042-016-4284-3

[92] M. Stéphane, *CHAPTER 8 - Wavelet Packet and Local Cosine Bases*. Boston: Academic Press, 2009, pp. 377–434. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780123743701000124

[93] K. Suwabe, M. Onuki, Y. Iizuka, and Y. Tanaka, "Globalized bm3d using fast eigenvalue filtering," in *2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Dec 2015, pp. 438–442.

[94] Y. Tian and Z. Wang, "An adaptive orthogonal matching pursuit algorithm based on redundancy dictionary," in *Fuzzy Systems and Knowledge Discovery (FSKD), 2013 10th International Conference on*, July 2013, pp. 578–582.

[95] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Computer Vision, 1998. Sixth International Conference on*, 1998, pp. 839–846.

[96] P. Tseng, "Convergence of a block coordinate descent method for nondifferentiable minimization," *J. Optim. Theory Appl.*, vol. 109, no. 3, pp. 475–494, Jun. 2001.

[97] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *Image Processing, IEEE Transactions on*, vol. 13, no. 4, pp. 600–612, April 2004.

[98] D. Wei, A. C. Bovik, and B. L. Evans, "Generalized coiflets: a new family of orthonormal wavelets," in *Signals, Systems amp; Computers, 1997. Conference Record of the Thirty-First Asilomar Conference on*, vol. 2, Nov 1997, pp. 1259–1263 vol.2.

[99] B. Weiss, "Fast median and bilateral filtering," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 519–526, 2006. [Online]. Available: http://doi.acm.org/10.1145/1141911.1141918

[100] N. Wiener, *Extrapolation, Interpolation, and Smoothing of Stationary Time Series*. The MIT Press, 1964.

[101] Y. Wu, B. Tracey, P. Natarajan, and J. P. Noonan, "James–stein type center pixel weights for non-local means image denoising," *IEEE Signal Processing Letters*, vol. 20, no. 4, pp. 411–414, April 2013.

[102] B. Xu, Y. Cui, B. Zuo, J. Yang, and J. Song, "Polarimetric sar image filtering based on patch ordering and simultaneous sparse coding," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 7, pp. 4079–4093, July 2016.

[103] J. Xu, J. Hu, and X. Jia, "A multistaged automatic restoration of noisy microscopy cell images," *IEEE Journal of Biomedical and Health Informatics*, vol. 19, no. 1, pp. 367–376, Jan 2015.

[104] L. P. Yaroslavsky, *Digital Picture Processing: An Introduction*, ser. Springer series in information sciences; 9., r. illustrated, Ed. Berlin ; New York : Springer-Verlag, 1985, no. 85014808.

[105] Y. Zhan, M. Ding, L. Wu, and X. Zhang, "Nonlocal means method using weight refining for despeckling of ultrasound images," *Signal Processing*, vol. 103, pp. 201 – 213, 2014, image Restoration and Enhancement: Recent Advances and Applications. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0165168413005070

[106] L. Zhang, W. Dong, D. Zhang, and G. Shi, "Two-stage image denoising by principal component analysis with local pixel grouping," *Pattern Recognition*, vol. 43, no. 4, pp. 1531–1549, Apr. 2010.

[107] L. Zhang, S. Vaddadi, H. Jin, and S. K. Nayar, "Multiple view image denoising," in *Computer Vision and Pattern Recognition*, 2009, pp. 1542–1549.

[108] Y. Zhang, J. Liu, M. Li, and Z. Guo, "Joint image denoising using adaptive principal component analysis and self-similarity," *Information Sciences*, vol. 259, pp. 128 – 141, 2014. [Online]. Available: //www.sciencedirect.com/science/article/pii/S0020025513005458

[109] H. Zhong, J. Zhang, and G. Liu, "Robust polarimetric sar despeckling based on nonlocal means and distributed lee filter," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 7, pp. 4198–4210, July 2014.

[110] X. Zhu and P. Milanfar, "Automatic parameter selection for denoising algorithms using a no-reference measure of image content," *IEEE Transactions on Image Processing*, vol. 19, no. 12, pp. 3116–3132, 2010.

Appendix A

Performance Results

The performance of the denoising algorithms with various noise levels (σ). The results are obtained by measuring the differences between the original images and the denoised images. The highest values of SSIM are highlighted by a **wavy under-bar bold** font, while the highest values of PSNR are highlighted with a **bold** font.

	σ	σ = 10		σ = 20		σ = 40		σ = 60	
	Method	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Barbara Image	Noisy	24.795	0.7285	19.764	0.5248	16.328	0.3365	14.924	0.2503
	Bilateral	19.760	0.7806	19.314	0.7473	15.435	0.4370	12.782	0.1250
	NL-Means	25.106	0.9286	21.594	0.8450	17.634	0.6711	15.584	0.5387
	K-SVD	27.200	0.9204	23.353	0.8547	19.435	0.7255	17.087	0.6050
	BM3D	**31.987**	0.9287	**30.244**	**0.8935**	**27.021**	**0.8075**	**24.962**	**0.7399**
	Non-iPPB	24.241	0.9224	21.527	0.8581	17.631	0.6962	15.645	0.5709
	It-PPB	23.524	0.9128	20.750	0.8556	17.926	0.7291	16.273	0.6114
	PGPCA	27.744	0.9239	23.873	0.8565	20.591	0.7518	18.636	0.6747
	PLPCA	28.884	**0.9340**	25.304	0.8817	21.766	0.7768	19.769	0.7182
	PHPCA	28.663	0.9324	25.035	0.8779	21.636	0.7837	19.554	0.7136

	σ	σ = 10		σ = 20		σ = 40		σ = 60	
	Method	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
House Image	Noisy	24.408	0.6177	19.658	0.3979	16.762	0.2431	15.424	0.1805
	Bilateral	26.867	0.8680	25.566	0.8004	18.486	0.3089	10.857	0.2504
	NL-Means	29.931	0.8836	26.872	0.8056	22.595	0.6400	19.864	0.4920
	K-SVD	30.294	0.8950	27.015	0.8285	23.143	0.7372	20.559	0.6374
	BM3D	29.861	0.9016	**28.380**	**0.8448**	**26.864**	**0.7952**	**25.960**	**0.7618**
	Non-iPPB	29.700	0.8854	27.205	0.8292	23.377	0.7202	20.812	0.6064
	It-PPB	29.144	0.8844	26.658	0.8283	23.684	0.7464	21.893	0.6576
	PGPCA	30.281	0.8983	27.114	0.8420	23.769	0.7586	21.826	0.7252
	PLPCA	**30.902**	**0.9031**	27.491	0.8404	23.800	0.7326	22.064	0.7139
	PHPCA	30.791	0.9012	27.484	0.8412	23.987	0.7573	22.026	0.7212
CurvedBand Image	Noisy	19.221	0.5012	15.068	0.2963	11.875	0.1752	10.465	0.1320
	Bilateral	34.645	0.9862	30.923	0.8872	21.083	0.2830	15.021	0.0978
	NL-Means	**37.916**	0.9631	32.627	0.8778	27.058	0.6670	23.570	0.5029
	K-SVD	35.287	0.9730	31.079	0.9383	26.702	0.8429	24.255	0.7578
	BM3D	35.080	**0.9869**	30.407	**0.9696**	26.082	**0.9339**	24.384	**0.9189**
	Non-iPPB	37.496	0.9785	**34.169**	0.9440	29.715	0.8258	25.760	0.6895
	It-PPB	36.731	0.9785	33.041	0.9508	**29.736**	0.8619	**27.213**	0.7557
	PGPCA	33.661	0.9640	29.573	0.9364	26.165	0.8592	24.457	0.8579
	PLPCA	34.526	0.9555	30.250	0.9220	26.362	0.7991	24.879	0.8261
	PHPCA	34.635	0.9612	30.171	0.9291	26.488	0.8444	24.705	0.8380
Chessboard Image	Noisy	15.513	0.5585	11.341	0.4397	8.139	0.3181	6.742	0.2503
	Bilateral	**47.898**	0.9853	37.586	0.9118	23.162	0.5564	15.662	0.3774
	NL-Means	25.050	0.9760	24.319	0.9435	22.771	0.8837	20.794	0.8371
	K-SVD	38.509	0.9609	32.666	0.8973	26.796	0.7995	23.076	0.7272
	BM3D	42.827	**0.9898**	**38.422**	**0.9689**	**33.357**	**0.9300**	**28.304**	**0.9033**
	Non-iPPB	21.668	0.9701	21.442	0.9407	20.398	0.8745	18.887	0.8280
	It-PPB	21.073	0.9601	20.676	0.9254	19.554	0.8631	18.216	0.8160
	PGPCA	34.036	0.9759	29.664	0.9537	23.812	0.8334	19.584	0.7675
	PLPCA	33.467	0.9411	28.836	0.9059	23.285	0.8150	20.276	0.7968
	PHPCA	33.408	0.9421	29.288	0.9227	23.893	0.8332	20.220	0.7873

Appendix B

Execution Time

Execution time in seconds shows the speed of the denoising methods applied to the four images at various noise levels (σ). The fastest patch-based methods are highlighted by a **wavy under-bar bold font**. In both tables, the methods are sorted from the oldest to the most recent.

	σ	σ = 10	σ = 20	σ = 40	σ = 60	Average
	Method	per second	per second	per second	per second	per second
Barbara Image	AD	0.05	0.05	0.05	0.05	0.05
	Bilateral	0.48	0.48	0.48	0.49	0.48
	NL-Means	44.13	46.16	43.66	46.58	45.13
	K-SVD	1125.85	1163.52	1116.98	1113.34	1129.92
	BM3D	**3.13**	**3.16**	**3.21**	**3.96**	**3.37**
	Non-it PPB	10.26	9.93	10.37	10.16	10.18
	It-PPB	32.10	28.60	30.23	28.68	29.90
	PGPCA	5.35	5.27	6.79	6.82	6.06
	PLPCA	9.42	9.05	15.43	16.99	12.72
	PHPCA	7.62	7.57	9.71	9.77	8.67

	σ	σ = 10	σ = 20	σ = 40	σ = 60	Average
	Method	per second	per second	per second	per second	per second
House Image	AD	0.05	0.05	0.05	0.05	0.05
	Bilateral	0.48	0.48	0.48	0.48	0.48
	NL-Means	44.23	45.50	43.76	43.45	44.24
	K-SVD	356.09	344.96	338.84	339.82	344.93
	BM3D	**0.84**	**0.82**	**0.80**	**0.98**	**0.86**
	Non-it PPB	10.97	10.02	10.35	10.10	10.36
	It-PPB	30.10	29.38	29.16	28.63	29.32
	PGPCA	1.32	1.34	1.65	1.65	1.49
	PLPCA	2.28	2.28	3.65	3.73	2.99
	PHPCA	1.76	1.71	2.15	2.12	1.93
CurvedBand Image	AD	0.05	0.05	0.05	0.05	0.05
	Bilateral	0.48	0.48	0.49	0.49	0.48
	NL-Means	42.84	45.93	43.42	45.83	44.50
	K-SVD	349.84	342.92	340.05	341.39	343.55
	BM3D	**0.32**	**0.82**	**0.84**	**0.99**	**0.74**
	Non-it PPB	9.88	9.99	9.95	10.06	9.97
	It-PPB	30.66	28.67	28.74	28.66	29.18
	PGPCA	1.34	1.39	1.63	1.68	1.51
	PLPCA	2.37	2.19	3.84	3.81	3.05
	PHPCA	1.75	1.74	2.19	2.13	1.95
Chessboard Image	AD	0.05	0.05	0.05	0.05	0.05
	Bilateral	0.47	0.48	0.48	0.47	0.48
	NL-Means	42.39	45.58	44.00	42.93	43.73
	K-SVD	1161.89	1155.13	1115.27	1112.00	1136.07
	BM3D	**0.79**	**0.32**	**0.30**	**0.43**	**0.46**
	Non-it PPB	13.18	10.33	10.01	10.05	10.89
	It-PPB	41.24	37.00	28.81	29.25	34.07
	PGPCA	0.79	0.86	1.03	0.98	0.91
	PLPCA	1.35	1.36	2.03	2.11	1.71
	PHPCA	1.10	1.04	1.32	1.32	1.20

Appendix C

Thresholding Levels and Operators

The best suggested thresholding levels and operators for structure and non-structure patches

σ	Patch Types	Thresholding Levels	Operators
10	Structure	(Column F) 16 levels of thresholding	2.0-3.5
	Non-structure	(Column F) 16 levels of thresholding	2.6-4.1
20	Structure	(Column F) 16 levels of thresholding	2.0-3.5
	Non-structure	(Column F) 16 levels of thresholding	2.6-4.1
30	Structure	(Column E) 8 levels of thresholding	2.2-2.9
	Non-structure	(Column F) 16 levels of thresholding	2.5-4.0
40	Structure	(Column C) 4 levels of thresholding	2.4-2.7
	Non-structure	(Column F) 16 levels of thresholding	2.5-4.0
50	Structure	(Column B) 2 levels of thresholding	2.6-2.7
	Non-structure	(Column F) 16 levels of thresholding	2.4-3.9
60	Structure	(Column B) 2 levels of thresholding	2.6-2.7
	Non-structure	(Column F) 16 levels of thresholding	2.2-3.7
70	Structure	(Column B) 2 levels of thresholding	2.6-2.7
	Non-structure	(Column F) 16 levels of thresholding	2.1-3.6
80	Structure	(Column B) 2 levels of thresholding	2.6-2.7
	Non-structure	(Column F) 16 levels of thresholding	2.0-3.5
90	Structure	(Column B) 2 levels of thresholding	2.6-2.7
	Non-structure	(Column F) 16 levels of thresholding	2.0-3.5
100	Structure	(Column B) 2 levels of thresholding	2.6-2.7
	Non-structure	(Column F) 16 levels of thresholding	2.0-3.5

Appendix D

Discrete Wavelet Transforms Performances

The performance of the *modified block batching 3D* algorithm with various wavelet families when denoising `Lena` image

#	Trans.	σ = 10			σ = 20			σ = 30		
		Step1	Step 2	T	Step1	Step2	T	Step1	Step 2	T
1	`bior1.1`	35.53	**35.92**	2.33	32.25	**33.01**	2.43	30.19	**31.22**	2.39
2	`bior1.3`	35.62	**35.93**	2.37	32.38	**33.02**	2.47	30.28	**31.23**	2.43
3	`bior1.5`	35.50	35.91	2.43	32.33	32.99	2.52	30.23	31.19	2.73
4	`bior2.2`	35.50	35.91	2.72	32.33	32.99	2.74	30.23	31.19	2.47
5	`bior2.4`	35.50	35.91	2.43	32.33	32.99	2.93	30.23	31.19	2.48
6	`bior2.6`	35.50	35.91	2.42	32.33	32.99	2.73	30.23	31.19	2.48
7	`bior2.8`	35.50	35.91	2.42	32.33	32.99	2.73	30.23	31.19	2.48
8	`bior3.1`	35.50	35.91	2.43	32.33	32.99	2.72	30.23	31.19	2.47
9	`bior3.3`	35.50	35.91	2.42	32.33	32.99	2.73	30.23	31.19	2.61
10	`bior3.5`	35.50	35.91	2.42	32.33	32.99	3.05	30.23	31.19	2.48
11	`bior3.7`	35.50	35.91	2.42	32.33	32.99	2.45	30.23	31.19	2.49
12	`bior3.9`	35.50	35.91	2.44	32.33	32.99	2.46	30.23	31.19	2.48
13	`bior4.4`	35.50	35.91	2.42	32.33	32.99	2.46	30.23	31.19	2.48
14	`bior5.5`	35.50	35.91	2.42	32.33	32.99	2.46	30.23	31.19	3.07
15	`bior6.8`	35.50	35.91	2.65	32.33	32.99	2.46	30.23	31.19	2.47
16	`coif1`	26.22	34.94	2.5	26.07	32.13	2.95	25.87	30.48	2.7
17	`coif2`	5.59	8.21	2.26	5.57	6.75	1.99	5.57	6.26	2.11
18	`coif3`	28.93	34.46	2.86	28.58	32.08	2.67	28.04	30.61	2.85

#	Trans.	σ = 10			σ = 20			σ = 30		
		Step1	Step 2	T	Step1	Step2	T	Step1	Step 2	T
19	coif4	5.58	8.22	2.49	5.57	6.75	2.17	5.56	6.26	2.26
20	coif5	23.92	33.00	3.23	23.96	30.66	2.91	23.99	29.33	3.58
21	db1	35.53	**35.92**	2.47	32.25	**33.01**	2.57	30.19	**31.22**	2.55
22	db10	5.59	8.22	2.51	5.57	6.74	2.11	5.57	6.25	2.35
23	db11	25.08	34.46	3.27	25.08	31.75	2.74	25.02	30.17	3.18
24	db12	5.59	8.21	2.45	5.57	6.73	2.16	5.56	6.24	2.37
25	db13	23.48	33.30	2.92	23.54	30.68	2.85	23.62	29.26	3.02
26	db14	5.59	8.23	2.41	5.57	6.75	2.23	5.57	6.26	2.32
27	db15	29.08	34.53	2.92	28.63	32.09	2.86	28.06	30.60	3.05
28	db16	5.58	8.22	2.49	5.57	6.74	2.3	5.56	6.25	2.46
29	db17	23.56	32.40	3.06	23.63	30.02	2.99	23.69	28.78	3.18
30	db18	5.58	8.32	2.55	5.57	6.78	2.36	5.56	6.27	2.44
31	db19	29.15	34.72	3.03	28.78	32.24	2.98	28.20	30.71	3.32
32	db2	5.59	8.46	2.03	5.57	6.84	1.87	5.57	6.30	2.01
33	db20	5.58	8.20	2.62	5.57	6.73	2.43	5.56	6.24	2.54
34	db3	23.18	32.67	2.9	23.21	30.14	2.55	23.27	28.82	2.66
35	db4	5.58	8.25	2.25	5.57	6.76	1.92	5.56	6.25	1.86
36	db5	28.59	34.36	2.94	28.21	31.98	2.55	27.75	30.52	2.6
37	db6	5.58	8.14	2.46	5.57	6.73	1.99	5.56	6.25	1.95
38	db7	23.81	32.91	3.09	23.83	30.55	3.35	23.87	29.22	2.83
39	db8	5.59	8.29	2.51	5.57	6.77	2.05	5.56	6.26	2.24
40	db9	24.44	33.80	3.09	24.59	31.32	2.7	24.68	29.88	3.06
41	dct	35.62	35.90	2.75	32.33	32.96	2.38	30.25	31.14	2.34
42	dmey	25.77	34.73	3.33	25.66	31.96	4	25.51	30.35	3.4
43	rbio1.1	35.53	**35.92**	2.34	32.25	**33.01**	2.36	30.19	**31.22**	2.42
44	rbio1.3	34.96	35.89	2.41	31.89	32.95	2.45	29.91	31.15	2.76
45	rbio1.5	34.67	35.86	2.48	31.71	32.91	2.51	29.76	31.11	2.85
46	rbio2.2	34.67	35.86	2.48	31.71	32.91	2.51	29.76	31.11	2.84
47	rbio2.4	34.67	35.86	2.91	31.71	32.91	2.51	29.76	31.11	2.82
48	rbio2.6	34.67	35.86	2.48	31.71	32.91	2.51	29.76	31.11	2.83
49	rbio2.8	34.67	35.86	2.48	31.71	32.91	2.5	29.76	31.11	2.82
50	rbio3.1	34.67	35.86	2.47	31.71	32.91	2.51	29.76	31.11	2.85
51	rbio3.3	34.67	35.86	2.48	31.71	32.91	2.51	29.76	31.11	2.84

#	Trans.	σ = 10			σ = 20			σ = 30		
		Step1	Step 2	T	Step1	Step2	T	Step1	Step 2	T
52	rbio3.5	34.67	35.86	2.47	31.71	32.91	2.51	29.76	31.11	2.88
53	rbio3.7	34.67	35.86	2.49	31.71	32.91	2.65	29.76	31.11	3.06
54	rbio3.9	34.67	35.86	2.48	31.71	32.91	2.51	29.76	31.11	2.89
55	rbio4.4	34.67	35.86	2.48	31.71	32.91	2.51	29.76	31.11	2.7
56	rbio5.5	34.67	35.86	2.48	31.71	32.91	2.51	29.76	31.11	3.15
57	rbio6.8	34.67	35.86	2.48	31.71	32.91	2.51	29.76	31.11	2.95
58	sym10	5.58	8.26	2.29	5.57	6.76	2.1	5.56	6.26	2.48
59	sym11	23.19	32.78	2.87	23.17	30.24	2.8	23.22	28.90	2.79
60	sym12	5.58	8.23	2.39	5.57	6.75	2.18	5.56	6.25	2.11
61	sym13	34.27	35.88	2.8	31.68	32.97	2.76	29.87	31.18	2.78
62	sym14	5.58	8.24	2.42	5.57	6.76	2.23	5.56	6.25	2.19
63	sym15	23.19	32.78	3	23.17	30.24	3.17	23.22	28.90	2.91
64	sym16	5.58	8.24	2.49	5.57	6.76	2.3	5.56	6.25	2.24
65	sym17	34.82	35.90	2.95	31.94	32.99	2.88	30.03	31.20	2.9
66	sym18	5.58	8.25	2.55	5.57	6.76	2.36	5.56	6.26	2.3
67	sym19	23.20	32.77	3.13	23.17	30.24	3.05	23.22	28.89	3.04
68	sym2	5.59	8.46	2.04	5.57	6.84	1.86	5.57	6.30	2.2
69	sym20	5.58	8.24	2.61	5.57	6.76	2.67	5.56	6.25	2.36
70	sym3	23.18	32.67	2.62	23.21	30.14	2.55	23.27	28.82	2.68
71	sym4	5.59	8.39	2.1	5.57	6.80	2.12	5.56	6.29	1.87
72	sym5	32.97	35.81	2.53	30.97	32.91	2.5	29.43	31.13	2.53
73	sym6	5.58	8.32	2.16	5.57	6.78	1.99	5.56	6.27	1.93
74	sym7	23.19	32.74	3.21	23.17	30.21	2.68	23.21	28.86	2.68
75	sym8	5.58	8.27	2.23	5.57	6.76	2.06	5.56	6.27	1.98
76	sym9	32.43	35.77	2.67	30.66	32.87	2.64	29.24	31.10	2.65

Curriculum Vitae

Name:	Monagi H. Alkinani
Education and Degrees:	The University of Western Ontario - London, Ontario 2012-2017 Ph.D. Computer Science
	The University of Western Ontario - London, Ontario 2009-2011 M.Sc. Computer Science
	King Abdulaziz University - Jeddah, Saudi Arabia 2003-2007 B.Ed. Computer Science
Related Work Experience:	Computer Science Lecturer Jeddah University, Jeddah, Saudi Arabia 2016-Present
	Graduate Teaching and Research Assistant The University of Western Ontario 2014-2017
Certificates:	Western Certificate in University Teaching and Learning Teaching Support Centre (TSC) at Western - London, Canada
	Western Certificate in Academic & Professional Communications Teaching Support Centre (TSC) at Western - London, Canada

Presentations:

Gave a presentation (A Modified Block Matching 3D Algorithm for Additive Noise Reduction) for 20 mins
5th International Symposium CompIMAGE'16
Niagara Falls, NY, USA, 2016

Gave a presentation (North Atlantic Right Whale Localization and Recognition Using Very Deep and Leaky Neural Network) for 20 mins
5th International Symposium CompIMAGE'16
Niagara Falls, NY, USA, 2016

Gave a presentation (Denoising Multi-view Image Using Non-local Means with SVD-based Measure) for 20 mins
University of Western Ontario Research in Computer Science conference (UWORCS'16)
London, ON, Canada, 2016

Gave a presentation (Non-Local Means for Stereo Image Denoising Using Structural Similarity) for 25 mins
International Conference of Image Analysis and Recognition (ICIAR'15)
Niagara Falls, ON, Canada, 2015

Gave a presentation (NL-Means for Stereo Image Denoising Using SSIM) for 20 mins
University of Western Ontario Research in Computer Science conference (UWORCS'15)
London, ON, Canada, 2015

Gave a presentation (One-Dimensional Compressor for 2D Image By Gathering Smartly Images Pixels Bits) for 20 mins
University of Western Ontario Research in Computer Science conference (UWORCS'14)
London, ON, Canada, 2014

Honours And Awards:

The Second Best Research Presenter Prize
(UWOCS' 2016) Conference at Western - London, Canada

King Abduallah Program Scholarship
Jeddah, Saudi Arabia, 2008-2016

Bachelor's Degree with the Second Class Honour
Jeddah, Saudi Arabia, April 2007

First Place in Alith Region Inventors Competition
Alith, Saudi Arabia, July 2001

Publications:

[1] Alkinani, M. H., El-Sakka, M. R.: *"Patch-based models and algorithms for image denoising: a comparative review between patch-based images denoising methods for additive noise reduction.*", EURASIP Journal on Image and Video Processing 2016 [Revised]

[2] Alkinani, M. H., El-Sakka, M. R.: *"A Modified Block Matching 3D Algorithm for Additive Noise Reduction*", Mathematics for Applications Journal, Vol. 5, No. 2, pp. 93-103, DOI 10.13164/ma.2016.07, 2016

[3] Alkinani, M. H., El-Sakka, M. R.: *"Denoising Multi-view Images Using Non-local Means with Different Similarity Measures.*" 13th International Conference of Image Analysis and Recognition (ICIAR 2016), Póvoa de Varzim, Portugal, pp 101-109, DOI 10.1007/978-3-319-41501-7_12, July 13-15, 2016

[4] Alkinani, M. H., El-Sakka, M. R.: *"Non-Local Means for Stereo Images Denoising using Structural Similarity*." 12th International Conference of Image Analysis and Recognition (ICIAR 2015), LNCS 9164, Springer International Publishing Switzerland, pp. 1–9, DOI: 10.1007/978-3-319-20801-5 6, 2015