An Adaptive Weighted Average (WAV) Reprojection Algorithm for Image Denoising

Halimah Alsarayhi  
*The University of Western Ontario*

Supervisor  
Mahmoud El-Sakka  
*The University of Western Ontario*

Graduate Program in Computer Science

A thesis submitted in partial fulfillment of the requirements for the degree in Master of Science  
© Halimah Alsarayhi 2019

Follow this and additional works at: [https://ir.lib.uwo.ca/etd](https://ir.lib.uwo.ca/etd)

*Part of the Theory and Algorithms Commons*

**Recommended Citation**  
[https://ir.lib.uwo.ca/etd/6196](https://ir.lib.uwo.ca/etd/6196)

This Dissertation/Thesis is brought to you for free and open access by Scholarship@Western. It has been accepted for inclusion in Electronic Thesis and Dissertation Repository by an authorized administrator of Scholarship@Western. For more information, please contact wlsadmin@uwo.ca.
Patch-based denoising algorithms have an effective improvement in the image denoising domain. The Non-Local Means (NLM) algorithm is the most popular patch-based spatial domain denoising algorithm. Many variants of the NLM algorithm have proposed to improve its performance. Weighted Average (WAV) reprojection algorithm is one of the most effective improvements of the NLM denoising algorithm. Contrary to the NLM algorithm, all the pixels in the patch contribute into the averaging process in the WAV reprojection algorithm, which enhances the denoising performance. The key parameters in the WAV reprojection algorithm are kept fixed regardless of the image structure. In this thesis, an improved WAV reprojection algorithm is proposed, where the patch size is assigned adaptively based on the image structure. The image structure is identified using an improved classification method that is based on the structure tensor matrix. The classification result is also utilized to improve the identification of similar patches in the image. The experimental results show that the denoising performance of the proposed method is better than that of the original WAV reprojection algorithm, as well as some other variants of the NLM algorithm.

**Keywords:** Denoising, patch-based, Weighted Average (WAV) reprojection algorithm, structure tensor, classification
Dedication

To my lovely parents

Aisha Alsurayhi, and Abdullah Alsurayhi

Without whom my success would be impossible
Acknowledgements

First and foremost, I would like to thank God Almighty for giving me the ability to complete my thesis successfully. Without his blessings, this achievement would not have been possible.

I would like to express my very great appreciation to Dr. Mahmoud El-Sakka, my thesis supervisor, for his professional guidance and valuable support. His willingness to give his time so generously has been very much appreciated.

My grateful thanks are also extended to my husband Khalid Nuli for his support and encouragement throughout my study.

Finally, I wish to thank my friend, Seereen Noorwali, for her assistance in my difficult times.
Contents

Abstract ii
Dedication iii
Acknowledgements iv
List of Figures viii
List of Tables xv
List of Abbreviations xvi

1 Introduction 1
  1.1 Motivation ....................................................... 2
  1.2 Thesis contribution ........................................... 3
  1.3 Thesis Outline ................................................. 4

2 Background 5
  2.1 Additive White Gaussian Noise .............................. 5
2.2 Noise Level Estimation ........................................ 6
  2.2.1 Fast Noise Level Estimation ................................. 7
  2.2.2 Patch-based Noise Estimation ............................... 8
  2.2.3 Statistical-based Noise Estimation ......................... 8
2.3 Image Denoising .............................................. 8
2.4 Patch-Based Denoising Algorithm .............................. 9
  2.4.1 Non-local Means Algorithm ................................... 10
  2.4.2 Adaptive Smoothing Parameter NLM ......................... 11
  2.4.3 Adaptive Patch Size NLM .................................... 14
  2.4.4 NLM with Improved Similarity Computation Method ....... 18
  2.4.5 Weighted Average Reprojection (WAV) ....................... 20
  2.4.6 Image Denoising with Block-Matching and 3D Collaborative
       Filtering (BM3D) ............................................. 24
2.5 Conclusion .................................................. 26

3 Methodology .................................................. 27
  3.1 Improved WAV Reprojection Algorithm ......................... 28
    3.1.1 The Improved Classification Method ....................... 28
    3.1.2 The Improved WAV Reprojection Method ................... 36
  3.2 Selecting the patch size ...................................... 37
  3.3 Conclusion .................................................. 47

4 Experimental Results and Analysis .......................... 49
  4.1 Noisy Images Generation ..................................... 49
4.2 Performance Evaluation ............................................. 50
  4.2.1 Peak Signal to Noise Ratio (PSNR) .......................... 50
  4.2.2 Mean Structure Similarity Index (MSSIM) .................. 52
4.3 Results and Discussion ............................................ 52
  4.3.1 Performance Analysis using PSNR ......................... 53
  4.3.2 Performance Analysis using MSSIM ....................... 56
  4.3.3 Visual Quality Comparison ................................. 59
  4.3.4 Comparison Using the Intensity Profile .................. 68
  4.3.5 Comparison with BM3D ..................................... 72
4.4 Summary .......................................................... 73

5 Conclusion and Future Work ....................................... 75
  5.1 Conclusion .................................................... 75
  5.2 Future Work .................................................. 76

Curriculum Vitae ..................................................... 83
List of Figures

2.1 The strategy of the NLM algorithm in finding the similar patches [4],
the weight of $q_1$, and $q_2$ is larger than the weight of $q_3$ as $q_1$, and $q_2$
have similar neighbourhood to the reference pixel $p$ . . . . . . . . . . 11

2.2 The improved classification results on Barbara image: (a) Original
image, (b) Noisy image $\sigma = 20$. (c) The structure tensor classification,
(d) The improved classification by Hu and Luo [17]. (Orange: texture
with little noise, green: medium region, light blue: texture with high
noise and dark blue is the flat region) . . . . . . . . . . . . . . . . . . 17

2.3 The three basic steps of the WAV reprojection algorithm [34] . . . . 21

2.4 The difference between the (a) centred patches and (b) the decentred
patches in the gathering step [34] (The red patch is the reference patch) 22

2.5 The Gaussian kernel and the Flat kernel . . . . . . . . . . . . . . . . 23

2.6 The difference between the influence zone in the centred patch (a),
and the decentred patch (b) [34] (The patch size $W = 5$, and the
searching window $R=15$) . . . . . . . . . . . . . . . . . . . . . . . . 24

2.7 The two steps of the BM3D algorithm [9]. . . . . . . . . . . . . . . . 25
3.1 The classification results using the structure tensor on Lena image: (The red color represents edge pixels, the blue color represents the smooth pixels, and the green color represents texture/noise pixels) (a) Noisy image $\sigma = 10$, (b) Classification of noisy image $\sigma = 10$, (c) Noisy image $\sigma = 20$, (d) Classification of noisy image $\sigma = 20$, (e) Noisy image $\sigma = 30$, (f) Classification of noisy image $\sigma = 30$, (g) Noisy image $\sigma = 40$, (h) Classification of noisy image $\sigma = 40$  . . . . . . . . . . . . . . . . . . 30

3.2 The improved classification results on Lena image: (The red color shows the edge pixels, the blue color shows the smooth pixels, and the green color shows texture/noise pixels) (a) Noisy image $\sigma = 10$, (b) Classification of noisy image $\sigma = 10$, (c) Noisy image $\sigma = 40$, (d) Classification of noisy image $\sigma = 40$, (e) Noisy image $\sigma = 70$, (f) Classification of noisy image $\sigma = 70$, (g) Noisy image $\sigma = 100$, (h) Classification of noisy image $\sigma = 100$  . . . . . . . . . . . . . . . . . . 33

3.3 The improved classification results on Window image image: (The red color shows the edge pixels, the blue color shows the smooth pixels, and the green color shows texture/noise pixels) (a) Noisy image $\sigma = 10$, (b) Classification of noisy image $\sigma = 10$, (c) Noisy image $\sigma = 40$, (d) Classification of noisy image $\sigma = 40$, (e) Noisy image $\sigma = 70$, (f) Classification of noisy image $\sigma = 70$, (g) Noisy image $\sigma = 100$, (h) Classification of noisy image $\sigma = 100$  . . . . . . . . . . . . . . . . . . 34
3.4 The classification results comparison on Barbara image: (a) Original image, (b) Noisy image $\sigma = 20$, (c) The improved classification by Hu and Luo [17]. (Orange: texture with little noise, green: medium region, light blue: texture with high noise and dark blue is the flat region), (d) Our classification method: (The red color shows the edge pixels, the blue color shows the smooth pixels, and the green color shows texture/noise pixels) ....... 35

3.5 The basic steps of our improved method ....... 36

3.6 The Mean PSNR values of smooth areas in 25 different natural scene images: (a) Noise $\sigma = 10$ to 50, (b) Noise $\sigma = 60$ to 100 ....... 39

3.7 The Mean PSNR values of edge areas in 25 different natural scene images: (a) Noise $\sigma = 10$ to 50, (b) Noise $\sigma = 60$ to 100 ....... 41

3.8 The Mean PSNR values of texture areas in 25 different natural scene images: (a) Noise $\sigma = 10$ to 50, (b) Noise $\sigma = 60$ to 100 ....... 43

3.9 The performance of different patch sizes around the edges on Lena image (noise $\sigma = 10$): (a) Original Image, (b) Denoised image using patch $5 \times 5$, (c) Denoised image using patch $7 \times 7$, (d) Denoised image using patch $9 \times 9$, (e) Denoised image using patch $11 \times 11$, (f) Denoised image using patch $13 \times 13$. ....... 45
3.10 The performance of different patch sizes on texture part from Lena image (noise $\sigma = 10$): (a) Original Image, (b) Denoised image using patch $5 \times 5$, (c) Denoised image using patch $7 \times 7$, (d) Denoised image using patch $9 \times 9$, (e) Denoised image using patch $11 \times 11$, (f) Denoised image using patch $13 \times 13$.

3.11 The smooth part from Lena image

4.1 The 10 images used in our experiments: (a) Lena image $512 \times 512$, (b) Man image $512 \times 512$, (c) Couple image $512 \times 512$, (d) Columbia image $480 \times 480$, (e) Barbara image $720 \times 850$, (f) Boats image $720 \times 576$, (g) House image $768 \times 512$, (h) Light House image $512 \times 768$, (i) Window image $768 \times 512$, (j) Woman image $512 \times 512$

4.2 The mean PSNR values of the 10 images set in 10 different noise levels

4.3 The PSNR values of Window image in 10 different noise levels

4.4 The mean SSIM values of the 10 image set in 10 different noise levels

4.5 The SSIM values of the Window image in 10 different noise levels

4.6 The visual comparison of Window image with noise $\sigma = 10$: (a) Original Image, (b) Noisy image, (c) Denoised image using NLM algorithm, (d) Denoised image using WAV reprojection algorithm, (e) Denoised image using Adaptive patch shape NLM algorithm, (f) Denoised image using our proposed method
4.7 The visual comparison of Window image with noise $\sigma = 30$: (a) Original Image, (b) Noisy image, (c) Denoised image using NLM algorithm, (d) Denoised image using WAV reprojection algorithm, (e) Denoised image using Adaptive patch shape NLM algorithm, (f) Denoised image using our proposed method .......................... 61

4.8 The visual comparison of Window image with noise $\sigma = 50$: (a) Original Image, (b) Noisy image, (c) Denoised image using NLM algorithm, (d) Denoised image using WAV reprojection algorithm, (e) Denoised image using Adaptive patch shape NLM algorithm, (f) Denoised image using our proposed method .......................... 62

4.9 The visual comparison of Window image with noise $\sigma = 70$: (a) Original Image, (b) Noisy image, (c) Denoised image using NLM algorithm, (d) Denoised image using WAV reprojection algorithm, (e) Denoised image using Adaptive patch shape NLM algorithm, (f) Denoised image using our proposed method .......................... 63

4.10 Zoomed part from Window image with noise $\sigma = 10$: (a) Original Image, (b) Noisy image, (c) Denoised image using NLM algorithm, (d) Denoised image using WAV reprojection algorithm, (e) Denoised image using Adaptive patch shape NLM algorithm, (f) Denoised image using our proposed method .......................... 64
4.11 Zoomed part from Window image with noise $\sigma = 30$: (a) Original Image, (b) Noisy image, (c) Denoised image using NLM algorithm, (d) Denoised image using WAV reprojection algorithm, (e) Denoised image using Adaptive patch shape NLM algorithm, (f) Denoised image using our proposed method

4.12 Zoomed part from Window image with noise $\sigma = 50$: (a) Original Image, (b) Noisy image, (c) Denoised image using NLM algorithm, (d) Denoised image using WAV reprojection algorithm, (e) Denoised image using Adaptive patch shape NLM algorithm, (f) Denoised image using our proposed method

4.13 Zoomed part from Window image with noise $\sigma = 70$: (a) Original Image, (b) Noisy image, (c) Denoised image using NLM algorithm, (d) Denoised image using WAV reprojection algorithm, (e) Denoised image using Adaptive patch shape NLM algorithm, (f) Denoised image using our proposed method

4.14 The two horizontal lines used for intensity profile

4.15 Difference in intensity profile when the NLM algorithm and our proposed method are applied along line $y = 230$

4.16 Difference in intensity profile when the WAV reprojection algorithm and our proposed method are applied along line $y = 230$

4.17 Difference in intensity profile when the adaptive patch shape NLM algorithm and our proposed method are applied along line $y = 230$

xiii
4.18 Difference in intensity profile when the NLM algorithm and our proposed method are applied along line $y = 450$. 

4.19 Difference in intensity profile when the WAV reprojection algorithm and our proposed method are applied along line $y = 450$. 

4.20 Difference in intensity profile when the adaptive patch shape NLM algorithm and our proposed method are applied along line $y = 450$. 

4.21 The PSNR performance of BM3D, the adaptive BM3D, and the proposed method on Lena image in 10 different noise levels.
List of Tables

3.1 The mean PSNR values of smooth areas in 25 different natural scene images using 10 different noise levels ............................................. 38
3.2 The mean PSNR values of edge areas in 25 different natural scene images using 10 different noise levels ............................................. 40
3.3 The mean PSNR values of texture areas (or noise) in 25 different natural scene images using 10 different noise levels ............................................. 42
3.4 The mean PSNR values of a part from Lena’s face and the background ............................................. 46

4.1 The mean PSNR values of the 10 images set in 10 different noise levels ............................................. 54
4.2 The PSNR values of Window image in 10 different noise levels ............................................. 55
4.3 The mean SSIM values of the 10 image set in 10 different noise levels ............................................. 57
4.4 The SSIM values of the Window image in 10 different noise levels ............................................. 58
4.5 The PSNR performance of BM3D, the adaptive BM3D, and the proposed method on Lena image in 10 different noise levels ............................................. 72
# List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLM</td>
<td>Non-Local Means</td>
</tr>
<tr>
<td>WAV</td>
<td>Weighted Average reprojection algorithm</td>
</tr>
<tr>
<td>BM3D</td>
<td>Block Matching and 3D collaborative filtering</td>
</tr>
<tr>
<td>DCT</td>
<td>Discrete Cosine Transform</td>
</tr>
<tr>
<td>FFT</td>
<td>Fast Fourier Transform</td>
</tr>
<tr>
<td>PSNR</td>
<td>Peak Signal to Noise Ratio</td>
</tr>
<tr>
<td>MSSIM</td>
<td>Mean Structural similarity Index</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

Digital images are represented as a two-dimensional array of pixels. So, each pixel
defined as $f(x, y)$, where $f$ is the grey level value or the pixel intensity, and $(x, y)$ are
the spatial coordinates of the image pixel. Digital images might be contaminated by
noise during acquisition or transmission processes, which affected the original image
signals. Image noise could cause problems in some other image processing stages
such as image segmentation or image recognition. Therefore, image denoising is an
important process to restore the original clean image signals from its observed noisy
signals.

Over the years, variant image denoising approaches have been proposed. Most of
them achieved denoising by averaging in a way or another. This averaging may be
performed in spatial domain such as Gaussian Smoothing model [15] [20], Anisotropic
Filtering (AF) [27] [37], and Total Variation minimization [32] [31]; or in the fre-
quency domain like Wavelet Thresholding methods [6] [29].
In recent years, patch-based denoising algorithm has drawn a lot of attention in the image denoising field, where the neighbouring patches (image block) in a specific search window participate in the denoising process for a certain reference patch in the noisy image. This helps to find the best estimate for the original image signal from its noisy signal.

1.1 Motivation

Patch-based denoising algorithms have become extremely popular in the image denoising field. They take the advantage of the similarity within the images, where image signals are restored by performing averaging between the similar patches in the image. Buades et al. [4] have introduced a patch-based algorithm called Non-Local Means (NLM) for image denoising.

Variants of NLM algorithm have been proposed to improve its performance by adaptively selecting some of the internal parameters. Some of these variants have assigned the smoothing parameter adaptively based on the image structure [7] [36] [39], or based on the noise level [41]. Some other variants are based on selecting the patch size adaptively using the image structure [17] [23] [40]. Besides the adaptive patch size, Deledalle et al [10] proposed a shape adaptive patches to address the problem of the halo of noise around the edges. Some other variants have improved the NLM algorithm by improving the method of computing the similarity between patches [18] [28] [38].
One of the significant improvements in the patch-based denoising methods is the WAV reprojection algorithm [34] which moved the reprojection method from the patch space to pixel space. The WAV reprojection algorithm takes the advantage of the whole patch, i.e., all pixels in the patch are exploited, which enhances the denoising performance.

The key parameters of the WAV reprojection algorithm have kept fixed, regardless of the image structure. So, one of our focus is to improve the performance of the WAV reprojection algorithm by assigning an adaptive patch size based on the image structure. Also, we propose to improve the WAV reprojection algorithm by improving matching the similar patches.

1.2 Thesis contribution

Edges are preserved better with a small patch size while smooth regions have better denoising performance with large patch size [39] [40]. In the WAV reprojection algorithm, the patch size has set to be fixed regardless of the image structure. So, we propose an adaptive patch size WAV reprojection algorithm that is based on the image structure. The image pixels are classified based on our improved classification method. We improve the classification method provided by [40] that is based on the eigenvalues of the structure tensor matrix. The improved method is derived by combining it with the discontinuity indicator [39] that classifies the image pixel into three regions. Also, a preprocessing step is added to the noisy image to reduce the
amount of noise that affected the classification procedure. We classified the image pixels into three classes: smooth, edge, texture / noise. The classification result is then used as a mask on the noisy image to assign the patch size adaptively.

Moreover, we used the classification scheme to improve the method of finding similar patches. Two patches consider being similar only if they are from the same class. Our proposed method has improved the original WAV reprojection algorithm in term of PSNR and SSIM performance. Also, it has better preserved edges and texture in the image.

1.3 Thesis Outline

Our thesis consists of five chapters. This chapter, Chapter 1, is an introductory chapter. In Chapter 2, we discuss in detail some patch-based image denoising algorithms. Our methodology is presented in Chapter 3 with some experiments to assign the best patch size for each class. In Chapter 4, we present the experimental results of our method, and we compare it with some other competitive methods. Chapter 5 concludes our thesis and states some future works.
Chapter 2

Background

Digital images are often affected by unwanted signals (noise) during the acquisition or transmission processes. Image noise can cause problems in further image processing processes such as identification, segmentation. Noise comes in different models. Different methods are used to handle each noise model. This chapter focuses on Additive White Gaussian Noise, which is the most common noise considered in image processing. Different denoising algorithms to address this type of noise are discussed.

2.1 Additive White Gaussian Noise

The additive noise occurs when the noise signal is added to the original image signal, it is generally modelled as:

\[ v(i) = u(i) + n(i), \]  

(2.1)
where \( v \) is the noisy image, \( u \) is the clean image, and \( n \) is the noise. In the Additive White Gaussian Noise, the noise has a uniform (constant) power over all frequencies with amplitude following the Gaussian distribution. So, for a Gaussian variable \( z \) the probability density function is:

\[
p(z) = \frac{1}{\sqrt{\pi \sigma}} e^{-\frac{(z-\mu)^2}{2 \sigma^2}}
\]

(2.2)

where \( \mu \) is the mean value, \( \sigma \) is the standard deviation, and \( \sigma^2 \) is the variance of \( z \), and \( z \) is the amount of noise.

### 2.2 Noise Level Estimation

There are different methods to estimate the noise variance. They generally classified as filtered based, patch-based, and statistical based. The filtered based approaches estimate the noise by a pre-filter process using a low pass filter to suppress the image structure [35] [19]. Then, the noise is estimated as the difference between the filtered image and the noisy one. The patch-based noise estimation decomposes the image into patches. Then, the homogeneous patches are used to estimate the noise level [24] [25]. The main issue in patch-based methods is how to specify the homogeneous patches. Also, some patch-based methods require high computation load as it estimates the noise in an iterative way. The statistical methods analyse the local variance distribution. Their results are more robust as the statistical methods used are insensitive to outliers [2] [14].

In our research, we just want to specify the noise range (low, or high). We have
used one of the filtered-based noise level estimations. It is just a simple method that
doesn’t require any high computation load (See Section 3.2).

2.2.1 Fast Noise Level Estimation

The fast noise level estimation [19] includes two steps, convolution and averaging.
The following Laplacian mask is applied first on the noisy image.

\[
N = \begin{bmatrix}
1 & -2 & 1 \\
-2 & 4 & -2 \\
1 & -2 & 1 \\
\end{bmatrix}
\]

The noise estimation is the mask operation using the mask \( N \), which has zero
mean and variance \( (4^2 + 4 \times (2)^2 + 4 \times (1)^2)\sigma_n^2 = 36\sigma_n^2 \). The variance of the
convolution of the \( N \) on the noisy image \( I \) gives an estimate of \( 36 \sigma_n^2 \) at each pixel.
So, the averaging is applied to get an estimated noise variance \( \sigma_n^2 \) as follow:

\[
\sigma_n^2 = \frac{1}{36(W-2)(H-2)} \sum_{image I} (I(x,y) * N)^2
\]  \hspace{1cm} (2.3)

where \( W \) and \( H \) are the width and the height of the image, respectively. \( (I(x,y) * N) \)
is the value of applying the mask \( N \) at position \( (x,y) \)
2.2.2 Patch-based Noise Estimation

The homogenous patches are targeted in the patch-based noise estimation [24] [25]. So, the image is first decomposed into a number of patches. The homogeneous patches are then distinguished as the patches with the smallest standard deviation because they have the least change of intensity among decomposed patches. As intensity variation of the homogeneous patches is mostly caused by the noise, the noise level is estimated from the selected patches. Patch-based noise estimation methods overestimate the noise level in the small noise levels, and they underestimate the noise in the large noise levels.

2.2.3 Statistical-based Noise Estimation

The analysis of the local variance distribution is used to estimate the noise in the statistical-based method [2]. The image is decomposed into a number of small blocks. Then, the local intensity variance of each block is calculated. The variance of all blocks are averaged to estimate the noise variance. The size of the blocks used should be as small as possible (e.g. $2 \times 2$).

2.3 Image Denoising

Image denoising is an important process to restore the original image signals from the noisy ones. The main objective in image denoising is to reduce noise while preserving edges and textures. Over the years, various image denoising approaches have been proposed. Most of
them achieved denoising by averaging in a way or another. This averaging may be performed in the spatial domain, or in the frequency domain. In the spatial domain, the spatial neighbouring pixels are considered in denoising, such as the convolution of the Gaussian kernel [15]. In the frequency domain, the image is transformed to the frequency domain like Fourier transform [8], or Wavelet transform [26]. The filtering is performed in the frequency domain, then the inverse of the transform is applied to get the denoised image.

Recently, patch-based denoising algorithms have became extremely popular in the denoising field. They take the advantage of the similarity within the images. So, the averaging is performed based on the similarity between image patches. Buades et al. [4] have introduced the patch-based method for image denoising. They developed the NLM (Non-Local Means) algorithm. Other patch-based denoising algorithm that has the best performance results in denoising is BM3D [9]. In this chapter, various patch-based denoising algorithms are discussed.

2.4 Patch-Based Denoising Algorithm

Natural scene images have a high degree of similarity, which means that each small block has many similar blocks in the same image. Patch-based denoising algorithms restore image signals by exploiting this similarity.
2.4.1 Non-local Means Algorithm

The (NLM) algorithm [4] estimates the original image signal by considering the neighbouring area of the current processed pixel. For each pixel, they find all nearby similar patches in the image that match the patch around the current processed pixel (See Figure 2.1). Similar patches are chosen under some conditions based on the similarity in the gray-level value, the geometrical configuration in a whole neighbourhood, and how far they are from the current pixel. Then, the weighted average of all centred pixels in all similar patches is calculated. Those weights depend on the similarity between the patches. That is, pixels that their surroundings are similar in the gray-level values to the current processed one have more weights than other pixels, and closer pixels have more weight than the faraway one.

For a noisy image \( v \), where \( v = \{v(i)|i \in I\} \), \( i \) is the coordinate of a pixel within the image, the NLM for a pixel at location \( i \),

\[
NL[v](i) = \sum_{j \in i} w(i, j)v(i) \tag{2.4}
\]

where \( w \) is the weight, and it depends on the similarity between pixels at locations \( i \) and \( j \). Euclidean distance has been used to calculate the distance between patches. Each weight \( w \) has to be less than or equal 1 and larger than or equal to 0, while the total of all weights has to equal 1. The weight is calculated as follow:

\[
w(i, j) = \frac{1}{Z(i)}e^{-\frac{|v(N_i) - v(N_j)|^2}{2\sigma^2}} \tag{2.5}
\]
where $Z(i)$ is the normalizing constant, and it is defined as:

$$Z(i) = \sum_j e^{-\frac{||v(N_i) - v(N_j)||^2_{2,a}}{h^2}}$$

(2.6)

where $v(N_i)$ and $v(N_j)$ are the intensity gray level vectors of the neighbourhood of pixels $i$ and $j$ respectively. $||v(N_i) - v(N_j)||^2_{2,a}$ is the Euclidean distance between patches surrounding pixels $i$ and $j$, $a$ is the standard deviation of the Gaussian kernel and it is greater than 0, and $h$ is the constant that control the decay of the exponential weight function (smoothing parameter).

![Diagram](image)

Fig. 2.1 – The strategy of the NLM algorithm in finding the similar patches [4], the weight of $q_1$ and $q_2$ is larger than the weight of $q_3$ as $q_1$ and $q_2$ have similar neighbourhood to the reference pixel $p$

### 2.4.2 Adaptive Smoothing Parameter NLM

The smoothing parameter $h$ of the Gaussian kernel has an important impact on the performance of the NLM algorithm. Assigning the same fixed smoothing parameter
for smooth and edges areas, degrades the denoising performance of the NLM algorithm. Denoising smooth areas with the NLM algorithm works better with a larger smoothing parameter while at edges and textures areas using smaller smoothing parameter produces better results. Thus, different methods have been proposed to adaptively select the smoothing parameter for each pixel based on the image features.

Chen and Yang [7] have proposed a method to select the smoothing parameter adaptively. In their method the smoothing parameter is assigned based on the local grey-level variance of the image pixel. The local grey-level variance identifies the image structure, it distinguishes between the noise and the image edge or texture. The local grey-level variance is provided by:

\[ \sigma^2_l(x, y) = \frac{1}{D^2 - 1} \sum_{i,j=-(D-1)}^{D-1} [I(x + i, x + j) - m_l(x, y)]^2 \]  

(2.7)

Where \( D \) is the size of the sliding window, and \( m_l \) is the local mean of the pixels in the window. Based on the resulted classification, a small smoothing parameter is assigned for pixels on edges, or texture areas, and a large smoothing parameter is assigned for pixels on smooth areas.

Verma and Pandy [36] provided another method to choose the smoothing parameter adaptively. They classify the image regions using the Grey relational analysis, which is related to the theory of the grey system [11]. The Grey relational analysis applied when partial information is missed. It finds the relation between one fac-
tor to all other factors in a system. The grade of the grey relations between image patches is calculated using Grey relational analysis. Then, an adaptive smoothing parameter is assigned for each pixel based on the resulted classification.

One more method to assign an adaptive smoothing parameter is based on the discontinuity indicator [39]. A novel discontinuity indicator is proposed to detect image structure; it also distinguishes between edges and noise. The discontinuity indicator is derived from the eigenvalues of the structure tensor matrix [22]. It is obtained as the difference between the two eigenvalues for each pixel.

The resulted discontinuity indicator is used to classify image pixels. If the discontinuity indicator is large, the pixel considered to be on edge. If it is small and the two eigenvalues are also small, the pixel considered to be on smooth region. The pixel is noise if the discontinuity indicator is small and the two eigenvalues are large. Hence, an adaptive smoothing parameter is chosen for each pixel based on the resulted classification.

Zhu et.al. [41] proposed an improved NLM by applying the NLM method twice. First, the noise variance is estimated using the weak textured patch noise estimation [24]. The weak textured patches are selected based on the image gradient. Then, the Principle Component Analysis (PCA) is applied on the selected patches to attain the estimated noise level. After that, the Non-Local means method is applied on the noisy image using an adaptive smoothing parameter based on the estimated noise variance to get the basic estimate. The final estimate is then obtained by applying the Non-Local means method with less smoothing parameter as much noise has been
removed in the first step.

**Adaptive NLM using Weight Thresholding**

Khan and El-Sekka proposed the NLM using Weight Thresholding [21]. Their method is performed as a two-step approach. The basic estimate is generated in the first step by thresholding the weights of the pixels within the searching area. All weights above the threshold value are unchanged, but the weights less than the threshold are assigned to zero, thus those pixels are removed from the weighted averaging process. In the second step, the weighted thresholding NLM is applied once again but with different smoothing strength. The threshold value is adaptively assigned based on the noise level.

### 2.4.3 Adaptive Patch Size NLM

**Adaptive patch size based on image structure**

The patch size has set to be fixed in the original NLM algorithm. Using a variable patch size has a significant improvement on the performance of the NLM method. Edges are preserved better with a small patch size while smooth regions have better denoising performance with large patch size. Various adaptive patch size NLM denoising algorithm are proposed to overcome the inefficacy of the traditional NLM.

Zeng et al. [40] have proposed an adaptive patch size NLM algorithm. They use the structure tensor to classify the image into smooth and texture regions. The difference between the two eigenvalues of the structure tensor is calculated. If the difference between the two eigenvalues is large, the pixel is considered to be on a smooth area
and a large patch size is set to estimate its original value. On the other hand, if the
difference is small, the pixel is considered to be on a texture area and hence a small
patch size is set to estimate its original value.

Hu and Luo [17] has also improved the NLM algorithm by adaptively select the patch
size based on image structure. They use the local geometry of the image and the
noise variance to drive a new metric,

\[ R(i) = f(i) \left( \frac{\lambda_1^2 - \lambda_2^2}{\lambda_1^2 + \lambda_2^2} \right) \]  

(2.8)

where \( f(i) \) is a feature detector based on the image histogram, \( \lambda_1 \) and \( \lambda_2 \) are
eigenvalues of the structure tensors that used to classify the image structure. This
composed metric classifies the image into four regions \((c_1, c_2, c_3, \text{ and } c_4)\) based on
their relations to three bin values \(T_1 = 90\%, T_2 = 70\%,\) and \(T_3 = 30\%\) (See Figure
2.2),

\[
i \in \left\{ \begin{array}{ll}
    c_1 & R(i) \geq T_1 \\
    c_2 & T_2 < R(i) < T_1 \\
    c_3 & T_3 < R(i) \leq T_2 \\
    c_4 & R(i) \leq T_3
\end{array} \right. 
\]  

(2.9)

where \( c_1 \) represents texture region with a small value of noise variance, \( c_2 \) is the
medium region that has even smaller noise variance and less structure, \( c_4 \) is the flat
region, and \( c_3 \) represents region with textures and high value of noise. Then, dif-
ferent patch sizes are applied to each region. The largest patch size is assigned to
the flat regions \( c_4 \), and the smallest patch size is assigned to \( c_3 \) as it has more tex-
ture. The patch size increases gradually in $c_1$ and $c_2$ as the regions have less texture.

Lan et al [23] have also proposed an adaptive patch size NLM scheme based on region homogeneity. In their method, the patch size and the searching window sizes are assigned adaptively depending on the local scale measure. The local scale finds the homogeneous regions in the image according to the image structure [33]. The homogeneity range is large inside regions and it becomes smaller on edges. So, the local scale is computed for each pixel, then the patch size and the searching window size are assigned accordingly.

**Adaptive Patch Shape NLM**

Besides the adaptive patch size, Deledalle et al [10] proposed a shape adaptive patches to address the problem of the halo of noise around the edges. Various shapes have been used including square, disk, pie, slices, and bands. The variety in shapes is applied to handle the geometrical structure of the image. Fast Fourier Transform (FFT) is used to deal with the different shapes of patches. For aggregation, the same weight is assigned for all patch shapes. The proposed algorithm has a noticeable improvement for the noise halo.

**The Patch Size and the Noise Variance**

The knowledge of the noise variance improves the performance of the NLM algorithm. The patch size is also affected by the noise variance. With the high noise variance,
Fig. 2.2 – The improved classification results on Barbara image: (a) Original image, (b) Noisy image $\sigma = 20$. (c) The structure tensor classification, (d) The improved classification by Hu and Luo [17]. (Orange: texture with little noise, green: medium region, light blue: texture with high noise and dark blue is the flat region)
using larger patches have better denoising results, i.e., larger patch sizes allow better discrimination between patches [13]. Thus, it is better to use large patch size for its robustness to noise. On the other hand, choosing large patch size prevents finding similarities for the small details, also it degrades the denoising performance around edges. Thus, the author suggests assigning an adaptive smoothing parameter $h$ with a global patch size to overcome the issue of smoothing the edges.

### 2.4.4 NLM with Improved Similarity Computation Method

The Euclidean distance with the geometric image structure are used to find similar patches. This method is based only on the noisy image, which might not provide the structure similarity efficiently, especially with high noise variance. Thus, other approaches have been provided to improve the similarity method.

**Curvelet based NLM algorithm**

In the curvelet based NLM [38], the Curvelet transform is first applied on the noisy image. The Curvelet transform is a directional multi-scale transform that produces an optimal representation of image features like edges [12]. Similar patches are determined based on the different levels of the resulted reconstructed images by the Curvelet transform beside the noisy image. The final denoised image is obtained by the weighted average of the centred pixels in similar patches.
K-Means Clustering based NLM

The K-means clustering is combined with NLM algorithm to improve matching the similar patches [28]. The K-Means algorithm classifies the image pixels into a number of clusters based on the distance of their intensities from the centroid [16]. First, the noisy image is smoothed using a Gaussian smoothing filter. Gaussian filtering [15] recovers the image signals by applying the convolution of the Gaussian kernel on the noisy image $I_0$.

$$I(x, y) = I_0(x, y) * G(x, y) \quad (2.10)$$

where

$$G(x, y) = \frac{1}{(4\pi h^2)} e^{-\frac{|x|^2}{4h^2}} \quad (2.11)$$

The K-means technique is then applied on the smoothed image $I$ to create the mask image $I_m$.

$$I_m = \sum_{i=1}^{K} \sum_{x_j \in s_i} \| x_j - \mu_i \|^2 \quad (2.12)$$

Where $\mu$ is the centroid or the mean of the cluster, and $k$ is the number of the clusters and it depends on the image structure. The resulted image $I_m$ is used as a mask for the noisy image to improve matching the similar patches. Finally, the denoised image is obtained as the weighted average of pixels in the similar neighbourhood $\Omega$ within the cluster $n$ as follows:

$$NL[v_n](i) = \sum_{j \in \Omega} w(i, j)v_n(i) \quad (2.13)$$
DCT-Based Non-Local Means

In order to improve the patch similarity, the spatial distance between patches is replaced by the distance between the discrete cosine transform (DCT) [1] coefficients in the frequency domain [18]. The neighbourhood patches are transformed from the spatial domain to the DCT frequency domain. Then, the DCT coefficients are obtained through the Zigzag scan. Consequently, the weight in the NLM method is replaced by:

\[
    w(i,j) = \frac{1}{Z(i)} e^{-\sum_{k=1}^{d} \frac{(C_d(N_i)_k - C_d(N_j)_k)^2}{2h^2}}
\]

(2.14)

where \( C_d(N_i)_k \) is the \( k \)th DCT coefficient of the neighbourhood of the subspace \( N_i \), \( d \in [1, M] \) where \( M \) is number of pixels in \( N_i \), \( Z \) is the normalization factor, and \( h \) is the Gaussian smoothing parameter.

2.4.5 Weighted Average Reprojection (WAV)

The Weighted Average (WAV) Reprojection algorithm is one of the significant improvements in the patch-based denoising methods. WAV reprojection algorithm [34] has improved the reprojection method from the patch space to pixel space. The denoising is performed in three basic steps: collecting the similar patches, performing the denoising for each patch, and reprojecting the denoised patch to the pixel domain (See Figure 2.3).
In the first step, the chi-squared distribution $\chi^2$ in conjunction with the Euclidean distance is used to test the similarity between patches. Two patches are considered to be similar if the Euclidean distance between them is less than the quantile of $\chi^2(W^2)$. To estimate a pixel $x$ after the grouping step, the weighted average of the different estimators of $x$ in all patches pixel $x$ is belong to is calculated.

$$\hat{I}_{Wav}(x) = \sum_{i=1}^{W^2} \beta_i \hat{P}_i(W^2 - i + 1)$$ (2.15)

The weight $\beta_i$ is based on minimizing variance between patches. Because WAV reprojection algorithm uses the flat kernel, $\beta_i$ is proportional to the number of patches used to estimate $\hat{P}_i$, and $\sum_{i=1}^{W^2} \beta_i = 1$.

Only the central pixel in each patch is used to estimate the current processed pixel in the original Non-local means algorithm, which degrades the performance of the denoising and creates the halo of noise around the edges. WAV reprojection algorithm
Fig. 2.4 – The difference between the (a) centred patches and (b) the decentred patches in the gathering step [34] (The red patch is the reference patch)

takes the advantage of the whole patch, i.e., all pixels in the patch are exploited, which enhances the denoising performance. The NLM algorithm performs poorly around edges because the patches centred on the edges have a few similar patches. The decentred patches is used in the WAV reprojection algorithm to enhance the denoising performance around the edges. The decentred patches have more similar patches than the centred patch, especially around the edges. Figure 2.4 shows the number of patches that could be considered to denoise the pixels near the edges.

The WAV reprojection algorithm allows a faster implementation for two reasons. First, it utilizes the flat kernel instead of the Gaussian Kernel (See Figure 2.5 ). With
the flat kernel all candidates have the same weight while Gaussian requires more calculations as the weight varies from candidate to another. The flat kernel has an advantage also in the denoising performance. it minimizes the problem encountered by the Gaussian kernel in NLM algorithm when the searching window is too large. More candidates are considered in the weight, and affects the impact of the good candidates. The flat kernel robustifies the estimator by hard thresholding the small coefficients.

The second reason is related to the searching window size, WAV reprojection algorithm has good results with a small searching window size, e.g., (searching window = 9). The influence zone is wider as the patches are decentred (See Figure 2.6). The influence zone in the centred patches with the searching window $R$ and patch size $W$ is $R + W - 1$. However, the decentred patches have a wider influence zone, it is $R + 2W - 2$. 

Fig. 2.5 – The Gaussian kernel and the Flat kernel
2.4.6 Image Denoising with Block-Matching and 3D Collaborative Filtering (BM3D)

BM3D [9] is one of the significant improvements in the patch-based denoising approach. It consists of two filtering steps; the hard thresholding filtering [5] that gives the basic estimate, and the Wiener filtering [30] which produces the final estimate (See Figure 2.7). Each step includes three operations: grouping, collaborative filtering, and aggregation.

In the grouping operation, each reference block is compared with other blocks in the image. The Euclidean distance is calculated, and the similar blocks are chosen if the distance is less than a specified threshold.

\[ S_{xR} = \{ x \in X | d(Z_{xR}, Z_x) < \tau_{match} \} \]  

(2.16)
Then, all matched blocks $S_{xR}$ are stacked to form a 3D array $Z_{SxR}$. A 3D transformation $\tau_{3D}$ is then applied on the 3D array $Z_{SxR}$ to create a sparse representation. The hard thresholding is applied on the transform coefficients, which resulted in multiple estimates for each pixel because the estimates may overlap. So, each estimate is given a weight which is inversely proportional to the number of the non-zero coefficients. In the aggregation operation, a weighted average of all estimates are calculated to attain the basic estimate of the image.

In the second filtering step, the Weiner filtering is applied on the stacked local estimates $\hat{Y}_{S_{xR}}$ that resulted from the hard thresholding step, and the stacked matched blocks from the noisy image. Then, the weighted average is calculated to get the final estimate of the image.

![Fig. 2.7 – The two steps of the BM3D algorithm [9].](image)

Although BM3D has an effective improvement on the patch-based denoising domain, it consumes more time as it involves two transformation steps.
BM3D using an Adaptive Thresholding

In order to improve the thresholding process, an adaptive hard-thresholding is assigned in the hard thresholding step [3] of the BM3D Method. The adaptive hard-thresholding is selected based on the geometric and luminance distance similarities between patches. When a patch is geometrically far from the reference patch, more thresholding is assigned to the patch transforming coefficients. The collaborative Weiner filtering step is also improved by assigning more weights for similar patches. They use weights for the 3D array patches. More weights are assigned for similar patches. PSNR is used as a patch similarity measure for selecting those weights.

2.5 Conclusion

This chapter discussed different patch-based denoising algorithms. Also, different variants of the NLM algorithm are described. They improved the NLM algorithm by the adaptation selection of some internal parameters in the NLM scheme, or by improving the similarity computation method. In addition, some other algorithms that have significant improvements in the patch-based domain are discussed. In our research we focus on the WAV reprojection algorithm. we studied its performance when utilizing image structures.
Chapter 3

Methodology

WAV reprojection algorithm is one of the most powerful patch-based denoising algorithms. It is a simple and effective method that performs the denoising in the spatial domain. Unlike the NLM algorithm, all the pixels in the patches are exploited in the WAV reprojection algorithm. A fixed patch size is used in the denoising process. The similarity between patches is calculated using the Euclidean distance between patches.

In our thesis, the patch size is assigned adaptively based on the image structure. The image structures are identified using the structure tensor matrix. It classifies the image pixels into three classes. In addition, the classification results are used to improve matching similar patches. Patches similar to the reference patch contribute into the averaging process if their central pixels are from the same class.
3.1 Improved WAV Reprojection Algorithm

The prior knowledge about the image structure helps to set the parameters properly. Edge information is preserved better with a small patch size, while smooth regions have better denoising performance with large patch size. In the WAV reprojection algorithm, the patch size has set to be fixed regardless of the image structure. In our research, we target setting the patch size adaptively based on the image structure to enhance the denoising performance. Thus, the image pixels need to be classified first. We have used our improved method to classify the image pixels. The next section explains in details our improved classification method.

3.1.1 The Improved Classification Method

The image pixels are first classified using the eigenvalues of the structure tensor matrix [22]. The structure tensor matrix is defined as follow:

\[ T_\sigma = \begin{bmatrix} j_{11} & j_{12} \\ j_{21} & j_{22} \end{bmatrix} = \begin{bmatrix} G_\sigma * (g_x(i,j))^2 & G_\sigma * g_x(i,j)g_y(i,j) \\ G_\sigma * g_y(i,j)g_x(i,j) & G_\sigma * (g_y(i,j))^2 \end{bmatrix} \]  

(3.1)

where \( g_x \) and \( g_y \) are the gradients information in x and y directions, and \( G_\sigma \) is the Gaussian kernel. Then, the two eigenvalues are calculated:

\[ \lambda_1 = \frac{1}{2} \left( j_{11} + j_{22} + \sqrt{(j_{11} - j_{22})^2 + 4j_{12}^2} \right) \]  

(3.2)

\[ \lambda_2 = \frac{1}{2} \left( j_{11} + j_{22} - \sqrt{(j_{11} - j_{22})^2 + 4j_{12}^2} \right) \]  

(3.3)
where \(j_{11} = G_{\sigma} \ast (g_x(i,j))^2\), \(j_{22} = G_{\sigma} \ast (g_y(i,j))^2\), and \(j_{12} = G_{\sigma} \ast g_x(i,j)g_y(i,j)\).

We follow the classification methods provided by [39] [40] to classify the image into three regions. The absolute difference between the two eigenvalues \(\lambda_1\) and \(\lambda_2\) is then calculated.

\[
\lambda = |\lambda_1 - \lambda_2| \tag{3.4}
\]

Then, the following classification scheme is used to classify image pixels:

\[
(i,j) \in \begin{cases} 
  c_1 & \lambda(i,j) \leq \lambda_2 \frac{(\lambda_1 - \lambda_2)}{n} \\
  c_2 & \lambda(i,j) \leq \lambda_2 \frac{2(\lambda_1 - \lambda_2)}{n} \\
  \vdots & \\
  c_n & \lambda(i,j) \leq \lambda_2 \frac{n(\lambda_1 - \lambda_2)}{n} 
\end{cases} \tag{3.5}
\]

This classification is inaccurate, as some pixels may belong to more than one class. Also, it fails to classify the image pixels in the high noise levels, Figure 3.1 shows the classification result, where the red color represents edge pixels, the blue color represents the smooth pixels, and the green color represents texture/noise pixels. In noise \(\sigma = 40\), we can notice the described method fails to define the image structure.

Thus, we propose to improve the classification in Equation 3.5 by combining it with the discontinuity indicator provided by Zeng et.al. [39]. The discontinuity indicator classify image pixels into smooth, edge and noise. If \(\lambda(i,j)\) is large, the
Fig. 3.1 – The classification results using the structure tensor on Lena image: (The red color represents edge pixels, the blue color represents the smooth pixels, and the green color represents texture/noise pixels) (a) Noisy image $\sigma = 10$, (b) Classification of noisy image $\sigma = 10$, (c) Noisy image $\sigma = 20$, (d) Classification of noisy image $\sigma = 20$, (e) Noisy image $\sigma = 30$, (f) Classification of noisy image $\sigma = 30$, (g) Noisy image $\sigma = 40$, (h) Classification of noisy image $\sigma = 40$
pixel is considered to be on an edge. If $\lambda(i, j)$ is small and the two eigenvalues are also small, the pixel is considered to be on a smooth region. The pixel is noise if $\lambda(i, j)$ is small but the two eigenvalues are large.

In our method, we classify the image pixels into three classes based on a comparison that made upon the two eigenvalues of the structure tensor matrix. We compare the two eigenvalues of each pixel in each resulted class from Equation 3.5 with a specified threshold value. If the two eigenvalues are smaller than the threshold, the pixel is considered to be in a smooth area. If the maximum eigenvalue $\lambda_1$ is larger than the threshold and the minimum eigenvalue $\lambda_2$ is smaller than the threshold, the pixel is considered on edge. The pixel is on texture or a noise if the two eigenvalues are larger than the threshold.

\[
(i, j) \in \begin{cases} 
    \text{Smooth} & \lambda_1 < \tau, \ \lambda_2 < \tau \\
    \text{Edge} & \lambda_1 > \tau, \ \lambda_2 < \tau \\
    \text{Texture/Noise} & \lambda_1 > \tau, \ \lambda_2 > \tau 
\end{cases} 
\tag{3.6}
\]

where $\tau$ is the threshold value, and it has set to be 40.

In addition, we apply a preprocessing step to improve the classification results. The image is denoised first using the original WAV reprojection algorithm. This step has improved the classification result, especially at the low noise levels. The texture areas can be classified as a third class when the noise level is less than 30. However, when the noise level is high, the third class represents noise. Moreover, our classification method has identified the image structure even when the noise level is high.
Figures 3.2, 3.3 present our classification results on two different natural scene images. One of them have mostly smooth regions (Lena), and the other one has more texture region (Window). The classification results are presented with different noise levels ($\sigma = 10$, $\sigma = 40$, $\sigma = 70$, $\sigma = 100$). The blue color presents the smooth areas, the red color presents the edges, and the green color presents the texture or noise areas. When the noise level is low ($\sigma = 10$), the green color shows the texture only. While texture and noise are presented in green color when noise level is high ($\sigma = 40$, $\sigma = 70$, and $\sigma = 100$). As the noise increase, the texture pixels tend to be presented as smooth pixels because the texture areas are blurred due to the denoising step. However, our method has identified the edges even if the noise level is high.

Our improved classification method has better results than the method described in Section 2.4.3. The noise signals could be distinguished from the original image signal, while in their method the regions classified as texture with low noise or texture with high noise. In the smooth region, if there is a noise, they just handle it as texture with high noise. Figure 3.4 shows the comparison between our classification method and the classification method proposed by Hu and Luo [17] on Barbara image.
Fig. 3.2 – The improved classification results on Lena image: (The red color shows the edge pixels, the blue color shows the smooth pixels, and the green color shows texture/noise pixels) (a) Noisy image $\sigma = 10$, (b) Classification of noisy image $\sigma = 10$, (c) Noisy image $\sigma = 40$, (d) Classification of noisy image $\sigma = 40$, (e) Noisy image $\sigma = 70$, (f) Classification of noisy image $\sigma = 70$, (g) Noisy image $\sigma = 100$, (h) Classification of noisy image $\sigma = 100$
Fig. 3.3 – The improved classification results on Window image: (The red color shows the edge pixels, the blue color shows the smooth pixels, and the green color shows texture/noise pixels) (a) Noisy image $\sigma = 10$, (b) Classification of noisy image $\sigma = 10$, (c) Noisy image $\sigma = 40$, (d) Classification of noisy image $\sigma = 40$, (e) Noisy image $\sigma = 70$, (f) Classification of noisy image $\sigma = 70$, (g) Noisy image $\sigma = 100$, (h) Classification of noisy image $\sigma = 100$
Fig. 3.4 – The classification results comparison on Barbara image: (a) Original image, (b) Noisy image $\sigma = 20$, (c) The improved classification by Hu and Luo [17]. (Orange: texture with little noise, green: medium region, light blue: texture with high noise and dark blue is the flat region), (d) Our classification method: (The red color shows the edge pixels, the blue color shows the smooth pixels, and the green color shows texture/noise pixels)
3.1.2 The Improved WAV Reprojection Method

Figure 3.5 shows the block diagram of the proposed scheme. After the classification step, we used the resulted classification as a mask on the noisy image. The improved method is then obtained as follow:

In the patchization step, patches similar to the reference patch contribute into the
averaging process only if their central pixels belong to the same class. That decreases
the number of un-similar patches from contributing in the averaging process.
In addition, an adaptive patch size is assigned to each pixel based on the class the
pixel is belong to. A large patch size is assigned to pixels on smooth areas, and a
small patch size is assigned to pixels on edges. For the texture, a smaller patch size
is assigned.

3.2 Selecting the patch size

In our experiments, we targeted natural scene images. We used 25 natural scene
images to select the best patch size for each class. The images are contaminated by
additive white Gaussian noise with 10 different levels of noise to assess the perfor-
mance of each class at each noise level and when using different patch sizes. Then,
the resulted mean PSNR values are used to assign the best patch size for each class.
We used patch sizes $5 \times 5$, $7 \times 7$, $9 \times 9$, $11 \times 11$, and $13 \times 13$. The searching window
size has set to be fixed as $9 \times 9$. The mean PSNR values of the 25 images in each
class are reported in Table 3.1, Table 3.2, and Table 3.3.
<table>
<thead>
<tr>
<th>Noise level</th>
<th>5 × 5</th>
<th>7 × 7</th>
<th>9 × 9</th>
<th>11 × 11</th>
<th>13 × 13</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>37.16</td>
<td>37.27</td>
<td><strong>37.31</strong></td>
<td>37.21</td>
<td>37.00</td>
</tr>
<tr>
<td>20</td>
<td>33.12</td>
<td>33.28</td>
<td>33.35</td>
<td><strong>33.36</strong></td>
<td>33.32</td>
</tr>
<tr>
<td>30</td>
<td>30.74</td>
<td>30.96</td>
<td>31.03</td>
<td><strong>31.04</strong></td>
<td>31.02</td>
</tr>
<tr>
<td>40</td>
<td>29.13</td>
<td>29.40</td>
<td>29.48</td>
<td><strong>29.50</strong></td>
<td>29.49</td>
</tr>
<tr>
<td>50</td>
<td>27.86</td>
<td>28.19</td>
<td>28.28</td>
<td><strong>28.30</strong></td>
<td>28.30</td>
</tr>
<tr>
<td>60</td>
<td>26.90</td>
<td>27.28</td>
<td>27.39</td>
<td><strong>27.41</strong></td>
<td>27.40</td>
</tr>
<tr>
<td>70</td>
<td>26.14</td>
<td>26.58</td>
<td>26.70</td>
<td><strong>26.72</strong></td>
<td>26.72</td>
</tr>
<tr>
<td>80</td>
<td>25.48</td>
<td>25.95</td>
<td>26.07</td>
<td><strong>26.10</strong></td>
<td>26.10</td>
</tr>
<tr>
<td>90</td>
<td>24.90</td>
<td>25.40</td>
<td>25.53</td>
<td><strong>25.56</strong></td>
<td>25.56</td>
</tr>
<tr>
<td>100</td>
<td>24.34</td>
<td>24.88</td>
<td>25.03</td>
<td><strong>25.06</strong></td>
<td>25.06</td>
</tr>
<tr>
<td><strong>Mean</strong></td>
<td>28.58</td>
<td>28.92</td>
<td>29.02</td>
<td><strong>29.03</strong></td>
<td>29.00</td>
</tr>
</tbody>
</table>

Table 3.1 – The mean PSNR values of smooth areas in 25 different natural scene images using 10 different noise levels

Figure 3.6 shows the bar chart of the mean PSNR comparison between the different patch sizes on the smooth areas. The results show that the smooth areas have the best mean PSNR performance in noise level 10 when the patch size of 9 × 9 is used (See Table 3.1). The patch size 11 × 11 have the best mean PSNR value in the noise levels from 20 to 60. In noise levels more than 70, both patch sizes 11 × 11 and 13 × 13 have the best performance, but the algorithm with small patch size executes faster. Hence, patch size 11 × 11 is selected for the pixels on the smooth areas.
Fig. 3.6 – The Mean PSNR values of smooth areas in 25 different natural scene images: (a) Noise $\sigma = 10$ to 50, (b) Noise $\sigma = 60$ to 100
Figure 3.7 plots the mean PSNR comparison when using different patch sizes on the edge areas. The PSNR values are also reported in Table 3.2. The results show that the patch size $5 \times 5$ has the best mean PSNR performance in noise level 10. Noise levels from 20 to 90 have the best results when the patch size of $7 \times 7$ is used. Patch $9 \times 9$ has the best results when the noise level is 100. The average in all noise levels shows that the best mean PSNR value for pixels on edges is when the patch $7 \times 7$ is used.

<table>
<thead>
<tr>
<th>Noise level</th>
<th>$5 \times 5$</th>
<th>$7 \times 7$</th>
<th>$9 \times 9$</th>
<th>$11 \times 11$</th>
<th>$13 \times 13$</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>32.81</td>
<td>32.62</td>
<td>32.22</td>
<td>31.85</td>
<td>31.58</td>
</tr>
<tr>
<td>20</td>
<td>28.46</td>
<td><strong>28.60</strong></td>
<td>28.54</td>
<td>28.34</td>
<td>28.11</td>
</tr>
<tr>
<td>30</td>
<td>25.70</td>
<td><strong>25.82</strong></td>
<td>25.77</td>
<td>25.64</td>
<td>25.46</td>
</tr>
<tr>
<td>40</td>
<td>23.81</td>
<td><strong>23.87</strong></td>
<td>23.78</td>
<td>23.63</td>
<td>23.44</td>
</tr>
<tr>
<td>50</td>
<td>22.47</td>
<td><strong>22.48</strong></td>
<td>22.37</td>
<td>22.21</td>
<td>22.02</td>
</tr>
<tr>
<td>70</td>
<td>20.98</td>
<td><strong>21.02</strong></td>
<td>20.93</td>
<td>20.79</td>
<td>20.64</td>
</tr>
<tr>
<td>80</td>
<td>20.68</td>
<td><strong>20.78</strong></td>
<td>20.72</td>
<td>20.61</td>
<td>20.49</td>
</tr>
<tr>
<td>90</td>
<td>20.49</td>
<td><strong>20.66</strong></td>
<td>20.64</td>
<td>20.56</td>
<td>20.47</td>
</tr>
<tr>
<td>100</td>
<td>20.41</td>
<td>20.65</td>
<td><strong>20.68</strong></td>
<td>20.64</td>
<td>20.57</td>
</tr>
<tr>
<td>Mean</td>
<td>23.74</td>
<td><strong>23.81</strong></td>
<td>23.71</td>
<td>23.56</td>
<td>23.39</td>
</tr>
</tbody>
</table>

Table 3.2 – The mean PSNR values of edge areas in 25 different natural scene images using 10 different noise levels
Fig. 3.7 – The Mean PSNR values of edge areas in 25 different natural scene images:
(a) Noise $\sigma = 10$ to 50, (b) Noise $\sigma = 60$ to 100
Table 3.3 shows the PSNR comparison for the texture/noise pixels for different patch sizes. Figure 3.8 presents the bar chart of the resulted mean PSNR values. The results show that the patch size of $5 \times 5$ has the best mean PSNR performance in noise levels less than or equal to 30. In noise level 40, patch size $9 \times 9$ has the best mean PSNR performance. In the noise levels between 50 and 90, the patch size of $11 \times 11$ has the best results. Both patch sizes $11 \times 11$ and $13 \times 13$ perform the same in the noise level 100.

<table>
<thead>
<tr>
<th>Noise level</th>
<th>$5 \times 5$</th>
<th>$7 \times 7$</th>
<th>$9 \times 9$</th>
<th>$11 \times 11$</th>
<th>$13 \times 13$</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>29.47</td>
<td>29.05</td>
<td>28.78</td>
<td>28.63</td>
<td>28.56</td>
</tr>
<tr>
<td>20</td>
<td>25.25</td>
<td>24.89</td>
<td>24.61</td>
<td>24.43</td>
<td>24.34</td>
</tr>
<tr>
<td>30</td>
<td>23.29</td>
<td>23.18</td>
<td>23.03</td>
<td>22.94</td>
<td>22.88</td>
</tr>
<tr>
<td>40</td>
<td>22.37</td>
<td>22.44</td>
<td>22.45</td>
<td>22.42</td>
<td>22.39</td>
</tr>
<tr>
<td>50</td>
<td>22.33</td>
<td>22.53</td>
<td>22.61</td>
<td>22.63</td>
<td>22.58</td>
</tr>
<tr>
<td>60</td>
<td>22.67</td>
<td>23.00</td>
<td>23.13</td>
<td>23.17</td>
<td>23.13</td>
</tr>
<tr>
<td>70</td>
<td>22.92</td>
<td>23.32</td>
<td>23.49</td>
<td>23.54</td>
<td>23.53</td>
</tr>
<tr>
<td>80</td>
<td>22.94</td>
<td>23.39</td>
<td>23.56</td>
<td>23.62</td>
<td>23.61</td>
</tr>
<tr>
<td>90</td>
<td>22.79</td>
<td>23.29</td>
<td>23.47</td>
<td>23.53</td>
<td>23.52</td>
</tr>
<tr>
<td>100</td>
<td>22.47</td>
<td>23.02</td>
<td>23.22</td>
<td>23.29</td>
<td>23.29</td>
</tr>
<tr>
<td>Mean</td>
<td>23.65</td>
<td>23.81</td>
<td>23.84</td>
<td>23.82</td>
<td>23.78</td>
</tr>
</tbody>
</table>

Table 3.3 – The mean PSNR values of texture areas (or noise) in 25 different natural scene images using 10 different noise levels
Fig. 3.8 – The Mean PSNR values of texture areas in 25 different natural scene images: (a) Noise $\sigma = 10$ to 50, (b) Noise $\sigma = 60$ to 100
To clearly show the effect of using different patch sizes on different areas of the image, we present an example that shows visually the denoising performance on different regions from Lena image, and when using different patch sizes. Figures 3.9, 3.10 show the effect of using a small patch size for edges or texture areas on zoomed parts from the Lena image. The noisy Lena image ($\sigma = 10$) is denoised using the WAV reprojection algorithm using different patch sizes. The results show how the artifacts are reduced significantly when using patch sizes $5 \times 5$ and $7 \times 7$. For the smooth region, smooth parts from Lena’s face and the background are cropped to test their performance (Figure 3.11). Table 3.4 presents the resulted PSNR values for each patch size on those areas. The results show that patch size $11 \times 11$ has the best PSNR values.

Therefore, we assigned patch size $11 \times 11$ for pixels on the smooth areas. Patch size $7 \times 7$ is assigning for pixels on edges. As the third class (texture) represents two types of pixels based on the noise level, we handle them differently. We assigned patch size $5 \times 5$ for noise levels less than or equal to 30, and patch size $11 \times 11$ for noise levels more than or equal to 30. Thus, the noise level of the noisy image needs to be estimated to assign the proper patch size for the third class. The noise level is estimated using the method described in Section 2.2.1. The result is then used to select the proper patch size for the third class.
Fig. 3.9 – The performance of different patch sizes around the edges on Lena image (noise $\sigma = 10$): (a) Original Image, (b) Denoised image using patch $5 \times 5$, (c) Denoised image using patch $7 \times 7$, (d) Denoised image using patch $9 \times 9$, (e) Denoised image using patch $11 \times 11$, (f) Denoised image using patch $13 \times 13$. 
Fig. 3.10 – The performance of different patch sizes on texture part from Lena image (noise $\sigma = 10$): (a) Original Image, (b) Denoised image using patch $5 \times 5$, (c) Denoised image using patch $7 \times 7$, (d) Denoised image using patch $9 \times 9$, (e) Denoised image using patch $11 \times 11$, (f) Denoised image using patch $13 \times 13$.

<table>
<thead>
<tr>
<th>Noise level</th>
<th>$5 \times 5$</th>
<th>$7 \times 7$</th>
<th>$9 \times 9$</th>
<th>$11 \times 11$</th>
<th>$13 \times 13$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Face</td>
<td>38.46</td>
<td>38.46</td>
<td>38.48</td>
<td><strong>38.49</strong></td>
<td>38.47</td>
</tr>
<tr>
<td>Background</td>
<td>38.82</td>
<td>38.82</td>
<td>38.83</td>
<td><strong>38.84</strong></td>
<td>38.82</td>
</tr>
</tbody>
</table>

Table 3.4 – The mean PSNR values of a part from Lena’s face and the background.
Summary of the Selected Patch sizes

Different patch size is assigned for each class in our method. The patch size, $w \times w$, is selected as shown below:

\[
w = \begin{cases} 
11, & \text{Smooth, (Texture/Noise ($\sigma > 30$))} \\
7, & \text{Edge} \\
5, & \text{Texture/Noise ($\sigma \leq 30$)} 
\end{cases} \quad (3.7)
\]

3.3 Conclusion

In this chapter, we explained our proposed method to improve the denoising performance of the WAV reprojection algorithm. First, the image pixels are classified...
using our improved classification method. The result is then used as a mask on the noisy image. An adaptive patch size is assigned for each pixel based on the class the pixels belong to. A patch size $11 \times 11$ is assigned for pixels on the smooth areas. For pixels on edges, patch size $7 \times 7$ is assigned. For the third class (texture and noise), $5 \times 5$ and $11 \times 11$ patch sizes are used based on the noise level. In addition, the process of matching patches has also improved. Patches are considered to be similar if the central pixel in each patch belong to the same class. The next chapter presents the experimental results of our proposed method.
Chapter 4

Experimental Results and Analysis

In our experiments, we used 10 natural scene images to compare our algorithm with other existing methods in the denoising domain. These images are Lena, Man, Couple, Columbia, Barbara, Boats, House, Light House, Window, Woman (See Figure 4.1). The images are then contaminated with additive white Gaussian noise with 10 different levels of noise from 10 to 100. We compare our algorithm with the original NLM algorithm, the WAV reprojection algorithm, adaptive patch shape NLM algorithm, and the state of art BM3D algorithm. The PSNR and the SSIM are used to compare the denoising performance. Also, we analyzed the performance using the intensity profile and the visual quality.

4.1 Noisy Images Generation

The additive white Gaussian noise signals are added to the original image signal to create a noisy image. In our experiments we produced the noisy image $v$ from the
clean image $u$ using the following equation in Matlab command:

$$v = u + \sigma \times \text{randn}(\text{size}(u))$$ (4.1)

where $\sigma$ is the standard deviation of the Gaussian noise, $\text{randn}$ is a function that generates a random matrix with Gaussian distribution, and $\text{size}$ is a function that used to get the dimension of the original input image.

### 4.2 Performance Evaluation

We have used the peak signal to noise ratio (PSNR) and the mean structure similarity index (MSSIM) to compare the denoising performance of our proposed method with the other existing methods in the denoising field. Those are the most common objective measures for the performance of the image denoising algorithms. Also, we did other subjective comparisons. We compare the visual quality and the intensity profile of our method with the other competitive methods.

#### 4.2.1 Peak Signal to Noise Ratio (PSNR)

The PSNR is the ratio between the maximum power of the original signal and the noise that affected its quality. The PSNR is defined via Mean Squared Error (MSE), which is calculated as the difference between the original image $u$ and the corrupted image $v$.

$$MSE = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} (v_{ij} - u_{ij})^2$$ (4.2)
Fig. 4.1 – The 10 images used in our experiments: (a) Lena image 512 × 512, (b) Man image 512 × 512, (c) Couple image 512 × 512, (d) Columbia image 480 × 480, (e) Barbara image 720 × 850, (f) Boats image 720 × 576, (g) House image 768 × 512, (h) Light House image 512 × 768, (i) Window image 768 × 512, (j) Woman image 512 × 512
where $M$ and $N$ are the dimensions of the images. Then, the PSNR is calculated as:

$$PSNR = 10 \log_{10} \left( \frac{(MAX)^2}{MSE} \right)$$ (4.3)

where $MAX$ is the maximum pixel intensity value. So, the PSNR is expressed in term of the logarithmic decibel scale. A larger value indicates better denoising.

4.2.2 Mean Structure Similarity Index (MSSIM)

The SSIM is a perceptual difference between two images. It has an advantage over the PSNR as it considers the similarity between various patches in the images and not only pixel by pixel. The SSIM between two patches $x$ and $y$ is defined as:

$$SSIM = \frac{(2\mu_x \mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$ (4.4)

where $\mu_x$ is the mean of $x$, $\mu_y$ is the mean of $y$, $\sigma_x^2$ is the variance of $x$, $\sigma_y^2$ is the variance of $y$, and $\sigma_{xy}$ is the covariance. $c_1$ and $c_2$ are two variables to stabilize the division. The SSIM produce a value between $[0, 1]$. A larger value indicates better denoising result. The mean SSIM (MSSIM), averaged over all SSIM, is used as a quality measurement of the denoising performance.

4.3 Results and Discussion

Our proposed method is compared with other denoising methods. It is compared based on the performance of the PSNR and the MSSIM. Also, some resulted images
are presented to show the visual performance.

4.3.1 Performance Analysis using PSNR

Our proposed method is compared with the other denoising algorithms, namely the original NLM, the WAV reprojection algorithm, and the adaptive patch shape NLM. The mean PSNR value of the 10 images are shown in Figure 4.2. The PSNR values are calculated in each noise level for each algorithm (Table 4.1). The values in *bold* present the best PSNR performance. Figure 4.2 compares the average PSNR between the proposed method and the other image denoising schemes.

The results show that our proposed method outperforms the other methods when the noise levels is less than 90. The mean PSNR value in all noise levels has the best result with our proposed method.

The PSNR values of the Window image Figure 4.1.(i) is also reported in Table 4.2. It shows the PSNR performance in 10 different noise levels for each algorithm. Figure 4.3 presents the PSNR comparison between our proposed method and the other image denoising methods on Window image. The results show that our proposed method has the best PSNR performance when the noise level is more than 20.
<table>
<thead>
<tr>
<th>Noise level</th>
<th>NLM</th>
<th>WAV</th>
<th>Adaptive shape NLM</th>
<th>Proposed method</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>33.28</td>
<td>33.41</td>
<td>33.89</td>
<td>33.93</td>
</tr>
<tr>
<td>20</td>
<td>29.67</td>
<td>30.31</td>
<td>30.41</td>
<td>30.45</td>
</tr>
<tr>
<td>30</td>
<td>27.50</td>
<td>28.20</td>
<td>28.19</td>
<td>28.30</td>
</tr>
<tr>
<td>40</td>
<td>26.02</td>
<td>26.61</td>
<td>26.57</td>
<td>26.82</td>
</tr>
<tr>
<td>50</td>
<td>24.93</td>
<td>25.43</td>
<td>25.39</td>
<td>25.65</td>
</tr>
<tr>
<td>60</td>
<td>24.08</td>
<td>24.54</td>
<td>24.52</td>
<td>24.76</td>
</tr>
<tr>
<td>70</td>
<td>23.40</td>
<td>23.85</td>
<td>23.86</td>
<td>24.07</td>
</tr>
<tr>
<td>80</td>
<td>22.84</td>
<td>23.30</td>
<td>23.33</td>
<td>23.49</td>
</tr>
<tr>
<td>90</td>
<td>22.35</td>
<td>22.84</td>
<td>22.89</td>
<td>22.97</td>
</tr>
<tr>
<td>100</td>
<td>21.93</td>
<td>22.42</td>
<td><strong>22.51</strong></td>
<td>22.50</td>
</tr>
</tbody>
</table>

| Mean       | 25.60| 26.09| 26.16             | **26.29**       |

Table 4.1 – The mean PSNR values of the 10 images set in 10 different noise levels

![Fig. 4.2 – The mean PSNR values of the 10 images set in 10 different noise levels](image-url)
<table>
<thead>
<tr>
<th>Noise level</th>
<th>NLM</th>
<th>WAV</th>
<th>Adaptive shape NLM</th>
<th>Proposed method</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>34.94</td>
<td>34.93</td>
<td>35.69</td>
<td>35.39</td>
</tr>
<tr>
<td>20</td>
<td>30.82</td>
<td>31.28</td>
<td>31.69</td>
<td>31.49</td>
</tr>
<tr>
<td>30</td>
<td>28.20</td>
<td>28.87</td>
<td>28.91</td>
<td>29.09</td>
</tr>
<tr>
<td>40</td>
<td>26.47</td>
<td>26.98</td>
<td>26.79</td>
<td>27.26</td>
</tr>
<tr>
<td>50</td>
<td>25.28</td>
<td>25.60</td>
<td>25.38</td>
<td>25.96</td>
</tr>
<tr>
<td>60</td>
<td>24.42</td>
<td>24.66</td>
<td>24.48</td>
<td>25.05</td>
</tr>
<tr>
<td>70</td>
<td>23.75</td>
<td>23.99</td>
<td>23.86</td>
<td>24.37</td>
</tr>
<tr>
<td>80</td>
<td>23.21</td>
<td>23.49</td>
<td>23.39</td>
<td>23.81</td>
</tr>
<tr>
<td>90</td>
<td>22.75</td>
<td>23.07</td>
<td>23.00</td>
<td>23.32</td>
</tr>
<tr>
<td>100</td>
<td>22.35</td>
<td>22.69</td>
<td>22.66</td>
<td>22.86</td>
</tr>
<tr>
<td><strong>Mean</strong></td>
<td><strong>26.22</strong></td>
<td><strong>26.56</strong></td>
<td><strong>26.58</strong></td>
<td><strong>26.82</strong></td>
</tr>
</tbody>
</table>

Table 4.2 – The PSNR values of Window image in 10 different noise levels

![Graph showing PSNR values for different noise levels](image_url)

Fig. 4.3 – The PSNR values of Window image in 10 different noise levels
4.3.2 Performance Analysis using MSSIM

The performance of our proposed method is compared in terms of MSSIM with the other denoising schemes. Table 4.3 presents the resulted MSSIM values for the 10 images set. Figure 4.4 compares the MSSIM values of our proposed method and the other denoising scheme.

When the noise level is less than 70, our method has the best MSSIM performance. The adaptive patch shape has the best results when the noise level is high. The average in all noise levels show that our proposed method has the best result.

The MSSIM performance of the Window image is also reported in Table 4.4, where our proposed method has the best MSSIM results when the noise level is more than 20. Figure 4.5 shows the MSSIM comparison of the proposed method and the other denoising methods.
<table>
<thead>
<tr>
<th>Noise level</th>
<th>NLM</th>
<th>WAV</th>
<th>Adaptive shape NLM</th>
<th>Proposed method</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.950</td>
<td>0.961</td>
<td>0.959</td>
<td>0.962</td>
</tr>
<tr>
<td>20</td>
<td>0.888</td>
<td>0.904</td>
<td>0.901</td>
<td>0.906</td>
</tr>
<tr>
<td>30</td>
<td>0.830</td>
<td>0.845</td>
<td>0.845</td>
<td>0.850</td>
</tr>
<tr>
<td>40</td>
<td>0.780</td>
<td>0.789</td>
<td>0.792</td>
<td>0.799</td>
</tr>
<tr>
<td>50</td>
<td>0.736</td>
<td>0.737</td>
<td>0.744</td>
<td>0.750</td>
</tr>
<tr>
<td>60</td>
<td>0.697</td>
<td>0.691</td>
<td>0.702</td>
<td>0.706</td>
</tr>
<tr>
<td>70</td>
<td>0.662</td>
<td>0.649</td>
<td>0.666</td>
<td>0.668</td>
</tr>
<tr>
<td>80</td>
<td>0.630</td>
<td>0.613</td>
<td>0.634</td>
<td>0.632</td>
</tr>
<tr>
<td>90</td>
<td>0.601</td>
<td>0.580</td>
<td>0.604</td>
<td>0.597</td>
</tr>
<tr>
<td>100</td>
<td>0.575</td>
<td>0.550</td>
<td>0.578</td>
<td>0.563</td>
</tr>
<tr>
<td>Mean</td>
<td>0.735</td>
<td>0.732</td>
<td>0.742</td>
<td>0.743</td>
</tr>
</tbody>
</table>

Table 4.3 – The mean SSIM values of the 10 image set in 10 different noise levels

![Graph showing mean SSIM values](image_url)

Fig. 4.4 – The mean SSIM values of the 10 image set in 10 different noise levels
<table>
<thead>
<tr>
<th>Noise level</th>
<th>NLM</th>
<th>WAV</th>
<th>Adaptive shape NLM</th>
<th>Proposed method</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.971</td>
<td>0.974</td>
<td>0.975</td>
<td>0.974</td>
</tr>
<tr>
<td>20</td>
<td>0.922</td>
<td>0.929</td>
<td>0.931</td>
<td>0.930</td>
</tr>
<tr>
<td>30</td>
<td>0.864</td>
<td>0.874</td>
<td>0.875</td>
<td>0.880</td>
</tr>
<tr>
<td>40</td>
<td>0.807</td>
<td>0.815</td>
<td>0.811</td>
<td>0.825</td>
</tr>
<tr>
<td>50</td>
<td>0.755</td>
<td>0.755</td>
<td>0.750</td>
<td>0.771</td>
</tr>
<tr>
<td>60</td>
<td>0.709</td>
<td>0.700</td>
<td>0.699</td>
<td>0.723</td>
</tr>
<tr>
<td>70</td>
<td>0.670</td>
<td>0.653</td>
<td>0.657</td>
<td>0.679</td>
</tr>
<tr>
<td>80</td>
<td>0.635</td>
<td>0.614</td>
<td>0.622</td>
<td>0.638</td>
</tr>
<tr>
<td>90</td>
<td>0.604</td>
<td>0.579</td>
<td>0.590</td>
<td>0.601</td>
</tr>
<tr>
<td>100</td>
<td>0.576</td>
<td>0.548</td>
<td>0.562</td>
<td>0.564</td>
</tr>
<tr>
<td><strong>Mean</strong></td>
<td><strong>0.751</strong></td>
<td><strong>0.744</strong></td>
<td><strong>0.747</strong></td>
<td><strong>0.758</strong></td>
</tr>
</tbody>
</table>

Table 4.4 – The SSIM values of the Window image in 10 different noise levels

![SSIM values graph](image-url)

Fig. 4.5 – The SSIM values of the Window image in 10 different noise levels
4.3.3 Visual Quality Comparison

It is important to assess the visual quality of the denoised image. The visual quality of our proposed method is compared with the other denoising algorithms. Figure 4.6, Figure 4.7, Figure 4.8, and Figure 4.9 show the visual comparison of the Window image when noise $\sigma = 10$, $\sigma = 30$, $\sigma = 50$ and $\sigma = 70$, respectively.

To better visualize the improvement, we zoomed in a part of the image. Figure 4.10, Figure 4.11, Figure 4.12, and Figure 4.13 present the zoomed images, respectively.

We can notice that our proposed method and the adaptive patch shape method have better attenuating the noise from the flower area when the noise level is 10, while WAV reprojection algorithm and the NLM algorithm produce artifacts on the same flower area.

In the noise $\sigma = 50$, and $\sigma = 70$, the zoomed parts show that our proposed method has better preserved the edges of the window frame area and the texture inside the window. The other methods have more blurry edges.
Fig. 4.6 – The visual comparison of Window image with noise $\sigma = 10$: (a) Original Image, (b) Noisy image, (c) Denoised image using NLM algorithm, (d) Denoised image using WAV reprojection algorithm, (e) Denoised image using Adaptive patch shape NLM algorithm, (f) Denoised image using our proposed method.
Fig. 4.7 – The visual comparison of Window image with noise \( \sigma = 30 \): (a) Original Image, (b) Noisy image, (c) Denoised image using NLM algorithm, (d) Denoised image using WAV reprojection algorithm, (e) Denoised image using Adaptive patch shape NLM algorithm, (f) Denoised image using our proposed method
Fig. 4.8 – The visual comparison of Window image with noise $\sigma = 50$: (a) Original Image, (b) Noisy image, (c) Denoised image using NLM algorithm, (d) Denoised image using WAV reprojection algorithm, (e) Denoised image using Adaptive patch shape NLM algorithm, (f) Denoised image using our proposed method
Fig. 4.9 – The visual comparison of Window image with noise $\sigma = 70$: (a) Original Image, (b) Noisy image, (c) Denoised image using NLM algorithm, (d) Denoised image using WAV reprojection algorithm, (e) Denoised image using Adaptive patch shape NLM algorithm, (f) Denoised image using our proposed method
Fig. 4.10 – Zoomed part from Window image with noise $\sigma = 10$: (a) Original Image, (b) Noisy image, (c) Denoised image using NLM algorithm, (d) Denoised image using WAV reprojection algorithm, (e) Denoised image using Adaptive patch shape NLM algorithm, (f) Denoised image using our proposed method
Fig. 4.11 – Zoomed part from Window image with noise $\sigma = 30$: (a) Original Image, (b) Noisy image, (c) Denoised image using NLM algorithm, (d) Denoised image using WAV reprojection algorithm, (e) Denoised image using Adaptive patch shape NLM algorithm, (f) Denoised image using our proposed method
Fig. 4.12 – Zoomed part from Window image with noise $\sigma = 50$: (a) Original Image, (b) Noisy image, (c) Denoised image using NLM algorithm, (d) Denoised image using WAV reprojection algorithm, (e) Denoised image using Adaptive patch shape NLM algorithm, (f) Denoised image using our proposed method
Fig. 4.13 – Zoomed part from Window image with noise $\sigma = 70$: (a) Original Image, (b) Noisy image, (c) Denoised image using NLM algorithm, (d) Denoised image using WAV reprojection algorithm, (e) Denoised image using Adaptive patch shape NLM algorithm, (f) Denoised image using our proposed method.
4.3.4 Comparison Using the Intensity Profile

The intensity profile on the Window image is used to compare the performance of our algorithm with other methods. Two different horizontal lines are used. Figure 4.14 shows the two lines that we used. The upper line is when \( y = 230 \), and the lower line is when \( y = 450 \). We calculated the absolute difference between the intensity profile of the original image and the denoised image. So, a larger value indicates a high change in the original image signal, and a small value indicates little change in the original image signal. To compare the performance of our proposed method with the other methods, we plot the resulted difference of our method with each method. Figures from 4.15 to 4.20 show the results. Our method is presented with the red color, while the other competitive methods are presented in blue color. Figure 4.15 and Figure 4.18 show that our proposed method has less error than the NLM algorithm. The red line is below the blue line in all cases. Our proposed method has less error than the original WAV reprojection algorithm (Figure 4.16 and Figure 4.19) and the adaptive patch shape (Figure 4.17 and Figure 4.20) in most cases.

Fig. 4.14 – The two horizontal lines used for intensity profile
Fig. 4.15 – Difference in intensity profile when the NLM algorithm and our proposed method are applied along line $y = 230$

Fig. 4.16 – Difference in intensity profile when the WAV reprojection algorithm and our proposed method are applied along line $y = 230$
Fig. 4.17 – Difference in intensity profile when the adaptive patch shape NLM algorithm and our proposed method are applied along line $y = 230$

Fig. 4.18 – Difference in intensity profile when the NLM algorithm and our proposed method are applied along line $y = 450$
Fig. 4.19 – Difference in intensity profile when the WAV reprojection algorithm and our proposed method are applied along line $y = 450$

Fig. 4.20 – Difference in intensity profile when the adaptive patch shape NLM algorithm and our proposed method are applied along line $y = 450$
4.3.5 Comparison with BM3D

The BM3D is considered to be the state of the art image denoising algorithm. Alk-inani and El-Sakka has improved its performance by an adaptive hard thresholding step [3]. Table 4.5 compares the PSNR values of BM3D, the adaptive hard threshold BM3D and our proposed method on Lena image. The results are also plotted in Figure 4.21. The adaptive hard threshold BM3D has the best PSNR values in noise levels from 10 to 30 and noise levels from 60 to 80, while BM3D has the best PSNR values in noise levels 40 and 50. Our proposed method has the best PSNR values in noise levels more than 80.

<table>
<thead>
<tr>
<th>Noise level</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
<th>60</th>
<th>70</th>
<th>80</th>
<th>90</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>BM3D</td>
<td>35.87</td>
<td>33.01</td>
<td>31.24</td>
<td>29.93</td>
<td>28.80</td>
<td>27.71</td>
<td>26.67</td>
<td>25.66</td>
<td>24.70</td>
<td>23.78</td>
</tr>
<tr>
<td>Adaptive BM3D</td>
<td>35.94</td>
<td>33.08</td>
<td>31.03</td>
<td>29.92</td>
<td>28.77</td>
<td>27.73</td>
<td>26.73</td>
<td>25.75</td>
<td>24.81</td>
<td>23.91</td>
</tr>
<tr>
<td>Proposed method</td>
<td>35.79</td>
<td>32.68</td>
<td>30.63</td>
<td>29.15</td>
<td>27.98</td>
<td>27.07</td>
<td>26.31</td>
<td>25.62</td>
<td>24.96</td>
<td>24.35</td>
</tr>
</tbody>
</table>

Table 4.5 – The PSNR performance of BM3D, the adaptive BM3D, and the proposed method on Lena image in 10 different noise levels.
Fig. 4.21 – The PSNR performance of BM3D, the adaptive BM3D, and the proposed method on Lena image in 10 different noise levels

4.4 Summary

Our proposed method has been compared with the NLM algorithm and some of its variants. We compared the performance subjectively and objectively. Our method outperforms the other NLM methods in term of PSNR when the noise level is more than 20. In MSSIM measure, it has the best results when the noise level is less than 73.
The visual comparison shows that our proposed method better preserved the edges and the texture in the image, especially in high noise levels. In the intensity profile, our method has mostly less change in the original image values than the other methods.
Chapter 5

Conclusion and Future Work

5.1 Conclusion

The WAV reprojection algorithm is one of the most significant improvements in the patch-based denoising algorithm. In this thesis, we have presented our improved method of the WAV reprojection algorithm. Our improved method includes the following:

- An improved classification scheme which based on the eigenvalues of the structure tensor matrix. It classifies the image pixels into three regions: smooth, edge, and texture/noise. The texture is presented clearly in the low noise levels, while the noise and textures are presented in the high noise levels.

- An adaptive patch size which assigned for each pixel based on the class the pixel is belong to. Some experiments are presented to select the best patch size for each class.
• The classification result is also used as a mask on the noisy image to improve grouping the similar patches. Only patches that their central pixels are from the same class contribute into the averaging process.

Our proposed method has compared with NLM algorithm and some of its variants in term of the PSNR and MSSIM. The results show that our proposed method has preserved edges better than the other competitive methods.

5.2 Future Work

Our proposed method could be extended to color images. So, the classification might even be improved because the luminance and the chrominance values would be used instead of the grey-scale values only. Improving the classification would improve the denoising performance.

Our proposed method could be applied also on different kind of data, like the medical images.
Bibliography


Curriculum Vitae

Name: Halimah Alsurayhi
Education and Degrees: Umm Al-Qura University, Mecca
From 2006 to 2012 B.Sc.
University of Western Ontario
London, ON
From 2016 to 2019 M.Sc.

Honours and Awards: Umm Al-Qura University research scholarship
From 2015 to 2022

Related Work and Experience:
Teaching Assistant
Umm Al-Qura University
From 2013 to 2015

Publications
Halimah Alsurayhi and Mahmoud El-Sakka. An Improved Weighted Average (WAV) Reprojection Image Denoising Method, 16th International Conference on Image Analysis and Recognition, 2019 (Accepted)