SPECTRAL GRAPH BASED IMAGE DENOISING METHODS

Master of Science Thesis

Ali Can YAĞAN

Eskişehir, 2016

SPECTRAL GRAPH BASED IMAGE DENOISING METHODS

Ali Can YAĞAN

MASTER OF SCIENCE THESIS

Graduate School of Sciences
Electrical and Electronics Engineering Program
Supervisor: Prof. Dr. Mehmet Tankut ÖZGEN

Eskişehir
Anadolu University
Graduate School of Sciences
December, 2016

This thesis work is supported in part by the Scientific Research Projects Commission of Anadolu University under the Master Thesis grant 1603F126.

FINAL APPROVAL FOR THESIS

This thesis titled “Spectral Graph Based Image Denoising Methods” has been prepared and submitted by Ali Can YAĞAN in partial fullfillment of the requirements in “Anadolu University Directive on Graduate Education and Examination” for the Degree of Master of Science in Electrical and Electronics Engineering Department has been examined and approved on 23/12/2016.

Committe Members		**Signature**
Member (Supervisor)	: Prof. Dr. Mehmet Tankut ÖZGEN	....
Member	: Prof. Dr. Levent ONURAL	....
Member	: Doç. Dr. Hakan Güray ŞENEL	....

....

Director

Graduate School of Science

ÖZET

SPEKTRAL ÇİZGE TABANLI GÖRÜNTÜ TEMİZLEME YÖNTEMLERİ

Ali Can YAĞAN

Elektrik-Elektronik Mühendisliği Anabilim Dalı
Anadolu Üniversitesi, Fen Bilimleri Enstitüsü, Aralık, 2016

Danışman: Prof. Dr. Mehmet Tankut ÖZGEN

Bazı spektral çizge tabanlı görüntü temizleme yöntemleri incelenmiş ve ayrıca bu metodların ulaştığı temizleme performanslarını geliştirmek için çizgesel Fourier alanında bir Wiener filtreleme işlemi önerilmiştir. Zaten ulaşılmış olan temizleme başarımını daha da iyileştirmek için bir işlem-sonrası basamağı olarak kullanılması önerilen Wiener filtresi, temizleme işleminden geçirilen çizgesel Fourier katsayıları kullanılarak tahmin edilmektedir. Bu çizgesel Wiener filtresi tüm görüntüden tahmin edilerek tüm görüntüye uygulanabilmekte ya da yerel uyarlanabilir bir şekilde görüntü parçalarına göre kullanılabilmektedir. Sonuçlarımız, çizgesel Fourier dönüşümünü hesaplamada kullanılan ağırlıklı yakınlık ve Laplasyen matrislerinin farklı seçenekleri için, ve elde edilen dönüşüm katsayılarını temizlemede kullanılan farklı işlemsel yöntemler için, önerilen Wiener filtresinin istikrarlı bir biçimde iyileştirme sağladığını göstermiştir. Basit yapıdaki bazı görüntüler için literatürdeki en iyi sonuçları veren yöntemlerden biri olan BM3D algoritmasından daha iyi PSNR değerleri elde edilmiştir.

Anahtar Kelimeler: Çizgelerde sinyal işleme, spektral çizge metodları, çizgesel Fourier dönüşümü, görüntü temizleme, Wiener filtresi.

ABSTRACT

SPECTRAL GRAPH BASED IMAGE DENOISING METHODS

Ali Can YAĞAN

Electrical and Electronics Engineering Program
Anadolu University, Graduate School of Sciences, December, 2016

Supervisor: Prof. Dr. Mehmet Tankut ÖZGEN

Some of the spectral graph based image denoising methods are reviewed and a Wiener filtering scheme in graph Fourier domain is proposed for improving image denoising performance achieved by these methods. The proposed Wiener filter is estimated by using graph Fourier coefficients of the noisy image after they are processed for denoising, to further improve the already achieved denoising accuracy as a post-processing step. It can be estimated from and applied to the entire image, or can be used patchwise in a locally adaptive manner. Our results indicate that the proposed step yields consistent accuracy improvement for different choices of weighted adjacency and graph Laplacian matrices used in computing the graph Fourier transform and for different processing methods used to denoise obtained transform coefficients. We obtain higher peak signal-to-noise ratio (PSNR) values than a state-of-the-art denoising method, known as the BM3D method, for some standard images.

Keywords: Signal processing on graphs, spectral graph methods, graph Fourier transform, image denoising, Wiener filter.

23/12/2016

STATEMENT OF COMPLIANCE WITH ETHICAL PRINCIPLES AND RULES

I hereby truthfully declare that this thesis is an original work prepared by me; that I have behaved in accordance with the scientific ethical principles and rules throughout the stages of preparation, data collection, analysis and presentation of my work; that I have cited the sources of all the data and information that could be obtained within the scope of this study, and included these sources in the references section; and that this study has been scanned for plagiarism with "scientific plagiarism detection program" used by Anadolu University, and that "it does not have any plagiarism" whatsoever. I also declare that, if a case contrary to my declaration is detected in my work at any time, I hereby express my consent to all the ethical and legal consequences that are involved.

....

Ali Can YAĞAN

ACKNOWLEDGMENTS

I would like to express my special appreciation to and thank sincerely my supervisor Prof. Dr. Mehmet Tankut ÖZGEN for his guidance, patience and support during my study. I could not imagine to have a better advisor than him.

I also would like to thank Prof. Dr. Levent ONURAL, Doç. Dr. Hakan Güray ŞENEL for serving in my committee. Their precious and insightful comments will be guidance to me for rest of my academic life.

I would like to thank each member of my loving family seperately for their endless trust and support.

I want to thank specially to Ms. Cansu SEVİK for providing an admirable support during my worst times. I always feel her presence and sympathy in my heart. I promise to be standing by her for my all life as long as she wants to be.

Ali Can YAĞAN

TABLE OF CONTENTS

Page

TITLE PAGE i

FINAL APPROVAL FOR THESIS ii

ÖZET iii

ABSTRACT iv

STATEMENT OF COMPLIANCE WITH ETHICAL PRINCIPLES AND RULES v

ACKNOWLEDGMENTS vi

TABLE OF CONTENTS vii

LIST OF FIGURES viii

LIST OF TABLES x

1. INTRODUCTION **1**

1.1. Overview and Motivation 1

1.2. Thesis Goals and Contributions 4

1.3. Thesis Outline 4

2. SPECTRAL GRAPH BASED IMAGE DENOISING **6**

2.1. Graph Weights 7

2.2. Denoising in the Graph Fourier Domain 8

2.3. Algorithms 10

3. WIENER FILTERING IN THE GRAPH FOURIER DOMAIN **12**

4. EXPERIMENTAL RESULTS **14**

5. CONCLUSION **26**

REFERENCES 27

APPENDIX 30

CURRICULUM VITAE

LIST OF FIGURES

Page

Figure 4.1. Original box image. 16

Figure 4.2. Noisy box image. 16

Figure 4.3. Denoised box image by the SGM3 algorithm. 16

Figure 4.4. Denoised box image by the SGM3W algorithm. 16

Figure 4.5. Denoised box image by the BM3D algorithm. 16

Figure 4.6. Original stripes image. 18

Figure 4.7. Noisy stripes image. 18

Figure 4.8. Denoised stripes image by the SGM1 algorithm. 18

Figure 4.9. Denoised stripes image by the SGM1W algorithm. 18

Figure 4.10. Denoised stripes image by the BM3D algorithm. 18

Figure 4.11. Original montage image. 20

Figure 4.12. Noisy montage image. 20

Figure 4.13. Denoised montage image by the SGM1 algorithm. 20

Figure 4.14. Denoised montage image by the SGM1W algorithm. 20

Figure 4.15. Denoised montage image by the BM3D algorithm. 20

Figure 4.16. Original cameraman image. 22

Figure 4.17. Noisy cameraman image. 22

Figure 4.18. Denoised cameraman image by the SGM1 algorithm. 22

Figure 4.19. Denoised cameraman image by the SGM1W algorithm. 22

Figure 4.20. Denoised cameraman image by the BM3D algorithm. 22

Figure 4.21. Original house image. 23

Figure 4.22. Noisy house image. 23

Figure 4.23. Denoised house image by the SGM1 algorithm. 23

Figure 4.24. Denoised house image by the SGM1W algorithm. 23

Figure 4.25. Denoised house image by the BM3D algorithm. 23

Figure 4.26. Original peppers image. 24

Figure 4.27. Noisy peppers image. 24

Figure 4.28. Denoised peppers image by the SGM1 algorithm. 24

Figure 4.29. Denoised peppers image by the SGM1W algorithm. 24

Figure 4.30. Denoised peppers image by the BM3D algorithm. 24

LIST OF TABLES

Page

Table 2.1. SGM1 Algorithm 10
Table 2.2. SGM2 Algorithm 10
Table 2.3. SGM3 Algorithm 11

Table 4.1. PSNR values (in dB) for SGM1, SGM1W, SGM2, SGM2W, SGM3, SGM3W and BM3D algorithms. 14
Table 4.2. PSNR values (in dB) for the stripes image for SGM1, SGM1W and BM3D algorithms for different noise standard deviation values. 15
Table 4.3. PSNR values (in dB) for different regions of the box image for SGM1, SGM1W algorithms. 17

1. INTRODUCTION

1.1. Overview and Motivation

Image processing is used in many areas such as security, criminal laboratory, military industry, underwater imaging, medicine (tomography, ultrasound), radar, photographic industry applications, physics, art and biomedicine. Image denoising is one of the main parts of image processing applications and has been a well-studied problem in the image processing community. As long as developing technology is a critical part of our lives, it seems this problem will continue to attract researchers with an aim to perform better restoration of images in the presence of noise. Especially with the increasing number of the pixels per image, modern image capturing devices become more sensitive to noise. However, in previous work on the analysis of the performance bounds for image denoising, it has been shown that there is still room for improvement for less textured, smoother images or for those without much structural complexity or variation [1]. Therefore, image denoising algorithms that are applied to reduce noise artifacts have still crucial importance for sensor and camera equipment manufacturers.

The most striking recent examples of denoising algorithms are kernel-based methods, where denoising is performed based on some type of shrinkage operation in a fixed or adaptive basis. The first modern adaptive method was the bilateral filter [2]. This edge-preserving smoothing method had the same spirit with the previous studies such as normalized convolution [3], Yaroslavky filter [4] and the SUSAN filter [5] which itself was an extention of the Yaroslavky filter. The common idea of these algorithms is weighting similar image pixels or data points (or more generally patches) locally or nonlocally [6], according to their affinities. However, identifying the similar data points under strong noise was a drawback of bilateral filter. This local filtering concept was extended to the entire image and a patch-averaging filter was introduced as nonlocal-means (NLM) filter [7,8]. This approach locates patches in accordance with their similarities and denoises the center pixel of a patch by taking weighted average of the center pixels of patches similar to it. In [9,10], a parameter-free algorithm was proposed to improve the NLM method by associating an adaptive neighborhood with a modified weight calculation formula.

A signal-dependent (data adaptive) steering kernel regression (SKR) framework was introduced in [11]. This method proved to be much more robust under strong noise.

A significantly different technique for denoising was first introduced as K-SVD algorithm in [12], that employs dictionary learning for denoising. In [13], a hybrid approach that merges dictionary-based and regression-based methods was proposed for better restoration of images.

In recent years, two best performing methods that define the current state-of-the-art were introduced. These methods employ Wiener filtering; one of them in a spatial patch domain [14], and the other one, which is the block-matching and three-dimensional filtering (BM3D) algorithm [15], in some three-dimensional transform domain such as a wavelet, discrete cosine and sine transforms, the Walsh-Hadamard transform, etc. However, BM3D takes a different approach to denoising. This method builds on the concept of NLM in identifying and grouping similar patches in an image, and performs the denoising process in transform domain as mentioned above.

Another approach related to kernel based filtering techniques covers spectral graph based image denoising methods that connect graph signal representation and Laplacian matrix in graph theory with non-local similarity in image processing. An image is viewed as a signal defined on a discrete, weighted, connected and undirected graph with vertices corresponding to image pixels in such methods [16–19]. Kernel similarity matrices are analogous to weighted adjacency matrices in spectral graph methods: A matrix element representing similarity between two image pixels serves as the weight of the graph edge connecting those two pixels (vertices) in such methods [6,18,19]. However, instead of applying row-normalized kernel matrices to input vectors composed of observed, noisy image pixels as in kernel based methods; in spectral graph methods, the noisy observed image vector is expanded in terms of the eigenvectors of the weighted adjacency matrix [20,21] or a graph Laplacian matrix derived from it [18], to obtain its graph Fourier transform (GFT) as the expansion coefficients [18,20]. Denoising process is performed in GFT domain by lowpass filtering arising from an l_2 regularization framework [18,19], or soft thresholding arising from l_1 regularization framework [21], or hard thresholding [22] of the graph Fourier coefficients. Then, the denoised image is obtained by computing the inverse GFT of the processed coefficient vector, i.e., multiplying it by the eigenvector matrix of the graph Laplacian or the weighted adjacency matrix [18, 20, 21].

The two-pass algorithm proposed in [22] uses the denoised image from the first pass to construct an improved graph for the second pass. In both passes, GFTs of images are computed by using eigenvectors of normalized graph Laplacian matrices and denoising is achieved by hard thresholding of graph Fourier coefficients. This work reports higher peak signal-to-noise ratio (PSNR) values than the BM3D method for a standard set of images, demonstrating that spectral graph based methods can compete with state-of-the-art image denoising techniques.

Elaborate spectral graph lowpass filter designs in GFT domain are developed in [23, 24], for image and graph signal denoising based on l_2 regularization involving more complex regularization operators in terms of graph Laplacian matrices, by minimizing l_2 regularized cost functions between observed, noisy and denoised images. Moreover, spectral graph Wiener filtering has been developed for stationary graph signals for denoising and deconvolution purposes [25–28]. However, underlying signals or images may be non-stationary or even be modelled as deterministic.

To our knowledge, a simple and more general implementation of a Wiener filter directly in graph Fourier domain, without requiring to estimate power spectral densities or correlation functions first, is lacking, for minimizing the mean-square error (MSE) between original and denoised images.

In [29], an estimated (MSE) is optimized with respect to truncation of eigenmodes of the denoising filter and iteration number while iteratively applying the filter, which yields a restricted optimization of filter eigenvalues instead of a true Wiener filtering operation.

To further improve denoising accuracies attained by such spectral graph based algorithms as reviewed above, we propose a Wiener filtering scheme in the GFT domain as a post-processing step. In this thesis, we use processed graph Fourier coefficients of a noisy image and the estimated noise variance to estimate graph Fourier coefficients of the underlying noiseless image and to construct a Wiener filter frequency response from them. The proposed Wiener filter can be applied either to the input noisy image or to the already denoised image. It can be estimated from and applied to the entire image, or can be used patchwise in a locally adaptive manner.

1.2. Thesis Goals and Contributions

Motivated by aforementioned facts, the goal of the thesis is to improve image denoising performance achieved by various spectral graph based image denoising methods, with the proposed Wiener filter post-processing step. Therefore, obtaining improvement for different kinds of benchmark images such as deterministic, less textured or complex structured etc. and for all spectral graph based image denoising methods is the main goal of this thesis. This thesis contributes to the literature in a few dimensions:

- The proposed post-processing Wiener filter is the first to be used for image denoising in spectral graph based methods.
- It can be estimated from and applied to the entire image, or can be used patchwise in a locally adaptive manner.
- It is promising that this proposed method appended to the spectral graph based algorithms can compete with the state-of-the-art image denoising techniques such as BM3D.

1.3. Thesis Outline

Organization of the thesis is as follows: Chapter 2 gives the background information about spectral graph based image denoising approaches that will be improved by our Wiener filter stage. In Section 2.1, different forms of graph weights are considered. Denoising methods in spectral graph domain are summarized in Section 2.2. Finally, spectral graph based image denoising algorithms considered in this work are presented in Section 2.3.

In Chapter 3, the proposed graph Wiener filter post-processing step is described in detail to further improve accuracies of spectral graph based image denoising methods mentioned in the previous chapter.

In Chapter 4, simulation results are presented to evaluate improvements achieved by the proposed approach and compare them with denoising accuracies of the BM3D algorithm for some benchmark images.

The thesis is concluded in Chapter 5.

2. SPECTRAL GRAPH BASED IMAGE DENOISING

In this chapter, we review some of the spectral graph based image denoising methods in the literature, and gather up three algorithms to be improved by appending our proposed Wiener filtering step to them. These algorithms to be improved are linear filtering algorithms in terms of input/output relationship.

In the work of Shuman et al. in [18], utilization of graph signals in many different engineering and science fields such as transportation networks, brain imaging, etc. is presented. In the paper, there are two developed ideas which are aimed at extending the application area of the graph signal processing. The first one is the idea of processing data on irregular data domains such as arbitrary graphs. Secondly, it is proposed to use well-known DSP algorithms on graph signals. As a result, a framework for processing data on graphs is achieved by adapting the elementary operators, such as filtering, convolution and translation, to graph setting. It has been shown that these approaches can be directly adapted to many operators, and it has several advantages due to the irregularity of the associated data.

To examine more elaborately, let X and Y be an original noiseless image and its noisy version with known noise variance, respectively, of size $m \times n$. Any such image can be regarded as a signal defined on a discrete, weighted, connected and undirected graph $G = (\mathcal{V}, E, K)$ that consists of a finite set of vertices (pixels) representing image pixels with cardinality $|\mathcal{V}| = mn = N$, a set of weighted edges $E \subseteq \mathcal{V} \times \mathcal{V}$ connecting vertices (pixels), and a weighted adjacency matrix K of size $N \times N$. If pixels i and j are connected via an edge $e = (i, j)$ directed from j to i, the element $K_{i,j}$ represents the positive weight of the edge as a measure of similarity between those pixels. If they are not connected, $K_{i,j} = 0$. Images can be considered as undirected graphs; hence $K_{i,j} = K_{j,i}$, i.e., K is a symmetric matrix [16–19].

The graph Laplacian is defined as

$$L = D - K$$

where the degree matrix D is diagonal matrix whose ith diagonal element is given by the sum of elements of the ith row of K. Since L is also a real symmetric matrix, it has a complete set of real, orthonormal eigenvectors $\mathbf{v_l}$, for $l = 0, 1, \ldots, N-1$.

Its eigenvalues are real and nonnegative, λ_l, for $l = 0, 1, \dots, N-1$, sorted from the smallest to the largest, with $\lambda_0 = 0$. Eigenvalues represent graph frequencies from the lower to the higher when sorted this way [16, 18].

The normalized graph Laplacian defined as [17, 18]

$$\tilde{L} = I - D^{-1/2} K D^{-1/2}$$

is also used in spectral graph based image denoising [22].

The image measurement model is given by

$$\mathbf{y} = \mathbf{x} + \mathbf{n}, \tag{2.1}$$

where $\mathbf{y}$ and $\mathbf{x}$ denote column-stacked vectorized forms of the observed noisy and original noiseless images, respectively, of size N. $\mathbf{n}$ is a noise vector of additive, zero-mean, white noise samples with variance σ^2. The image denoising problem is to estimate $\mathbf{x}$ given its noisy version $\mathbf{y}$.

2.1. Graph Weights

The observed image $\mathbf{y}$ is first pre-filtered by a Gaussian filter, in this thesis, and graphs used in denoising algorithms are constructed from the pre-filtered image $\tilde{\mathbf{x}}$ to stabilize graph weights by reducing the effect of noise, as suggested in [6, 7, 16, 19]. The following alternatives for the weighted adjacency matrix representing image pixel similarities are preferred among the others, in this work, because of their better results for benchmark images tested and presented in this thesis.

As the first alternative, the adjacency matrix is computed in a form reminiscent to that of the bilateral filter kernel [2, 6], as

$$K_{i,j} = \exp\left(-\|\tilde{\mathbf{x}}_\mathbf{i} - \tilde{\mathbf{x}}_\mathbf{j}\|^2 / h_x^2\right) \exp\left(-\|\mathbf{p}_\mathbf{i} - \mathbf{p}_\mathbf{j}\|^2 / h_p^2\right) \tag{2.2}$$

for $i, j = 1, 2, \dots, N$, where $\tilde{\mathbf{x}}_\mathbf{i}$ and $\tilde{\mathbf{x}}_\mathbf{j}$ denote vectorized forms of patches centered at pixels i and j in the pre-filtered image, and $\mathbf{p}_\mathbf{i}$ and $\mathbf{p}_\mathbf{j}$ denote their locations in the two-dimensional (2-D) image, respectively. The vector norm is Euclidean norm above. h_x and h_p are smoothing parameters in the photometric and spatial domains, respectively. Each pixel (vertex) is connected to each other in the image, with the above weight, in this first alternative.

In the second alternative graph structure, each image pixel is chosen to be connected to only its eight nearest neighbors in the 2-D image with the above weights again, rendering a sparse K matrix.

As a third alternative graph for representing images, each pixel is taken to be connected to only its eight nearest neighbors with weights as proposed in [22]:

$$K_{i,j}^{\mathrm{MS}} = \exp\left[-\left(\|\tilde{\mathbf{x}}_{\mathbf{i}} - \tilde{\mathbf{x}}_{\mathbf{j}}\| + \beta\|\mathbf{p}_{\mathbf{i}} - \mathbf{p}_{\mathbf{j}}\|\right)^2/\delta^2\right] \tag{2.3}$$

where the nonnegative parameter β controls the effect of patch similarity and the parameter δ controls the scaling of the overall similarity metric [22].

2.2. Denoising in the Graph Fourier Domain

Eigendecomposition of the symmetric graph Laplacian

$$L = D - K$$

or normalized graph Laplacian

$$\tilde{L} = I - D^{-1/2}KD^{-1/2}$$

obtained from the adjacency matrix K, is conducted first, as $L = VSV^T$. Then, orthonormal eigenvectors are used to compute the GFT of the observed noisy image vector $\mathbf{y}$ [18, 19, 22] as

$$\mathbf{y_f} = V^T\mathbf{y} \tag{2.4}$$

for denoising in the GFT domain, where $V = [\mathbf{v_0}, \mathbf{v_1}, \ldots, \mathbf{v_{N-1}}]$ denotes the eigenvector matrix. By processing the GFT of the noisy image, $\mathbf{y_f} = [y_f(\lambda_0), y_f(\lambda_1), \ldots, y_f(\lambda_{N-1})]^T$, the GFT of the denoised image

$$\hat{\mathbf{x}}_{\mathbf{f}} = [\hat{x}_f(\lambda_0), \hat{x}_f(\lambda_1), \ldots, \hat{x}_f(\lambda_{N-1})]^T$$

is obtained. As the last step, the inverse GFT is computed as

$$\begin{aligned} \hat{\mathbf{x}} &= V\hat{\mathbf{x}}_{\mathbf{f}} \\ &= \sum_{l=0}^{N-1} \hat{x}_f(\lambda_l)\mathbf{v_l} \end{aligned} \tag{2.5}$$

to yield the denoised image vector $\hat{\mathbf{x}}$.

The following denoising methods in the GFT domain are considered in this thesis. Denoising can be achieved by minimizing the l_2 regularized cost function,

$$\|\hat{\mathbf{x}} - \mathbf{y}\|^2 + \alpha \hat{\mathbf{x}}^T L \hat{\mathbf{x}}$$

leading to [18, 19]

$$\hat{\mathbf{x}} = (I + \alpha L)^{-1} \mathbf{y} = V(I + \alpha S)^{-1} V^T \mathbf{y}, \tag{2.6}$$

where the GFT of the noisy image vector is masked in the frequency domain by multiplying with a diagonal matrix and the inverse GFT of the result is computed to give the (vectorized) denoised image. Comparing (5) and (6) reveals the following lowpass filtering in the GFT domain [18, 19]

$$\hat{x}_f(\lambda_l) = \frac{1}{1 + \alpha \lambda_l} y_f(\lambda_l) \tag{2.7}$$

for $l = 0, 1, \ldots, N-1$. $\alpha > 0$ is a regularization parameter the value of which should be searched to result in a good denoising performance for a particular image, or for each patch of it if the above filter is applied patchwise.

Hard thresholding of graph Fourier coefficients of the noisy observed image is also used for denoising [22]:

$$\hat{x}_f(\lambda_l) = y_f(\lambda_l) \mathbf{1}(|y_f(\lambda_l)| \geq t), \tag{2.8}$$

for $l = 0, 1, \ldots, N-1$, where $\mathbf{1}(\cdot)$ returns 1 if the condition is satisfied and 0 otherwise. t is the threshold parameter to be searched for a good denoising accuracy, usually chosen to be proportional to the estimated standard deviation of the noise [7].

Assuming that the noiseless original image is sparse in the GFT domain, minimization of the l_1 regularized cost function

$$\|\mathbf{y} - V\hat{\mathbf{x}}_\mathbf{f}\|_2^2 + 2\alpha \|\hat{\mathbf{x}}_\mathbf{f}\|_1$$

leads to soft thresholding in the GFT domain;

$$\hat{x}_f(\lambda_l) = \max(0, |y_f(\lambda_l)| - \alpha)\,\mathrm{sgn}(y_f(\lambda_l)).$$

However, soft thresholding has resulted in worse denoising results than the other two methods, for algorithms and benchmark images considered. Therefore, in this thesis, soft thresholding was not included.

2.3. Algorithms

Our proposed Wiener filter will be appended to the following spectral graph based image denoising algorithms, designated as SGMs, to test the resulting improvement in accuracy for each of them.

Table 2.1. *SGM1 Algorithm*

```
SGM1 :
1. Set the graph weights ($K$) as given by (2.2);
2. Derive the Laplacian ($L = D - K$) and perform its eigendecomposition;
3. Compute the GFT of the observed noisy image $\mathbf{y_f}$ as given by (2.4);
4. Perform denoising process by lowpass filtering in the GFT domain as in (2.7);
5. Compute the denoised image $\hat{\mathbf{x}}$ by the inverse GFT as given by (2.5);
```

In SGM2, each image pixel is connected to only its eight nearest neighbors in the 2-D image, with the weights given by (2), to construct K. The remain steps are the same with those of $SGM1$.

Table 2.2. *SGM2 Algorithm*

```
SGM2 :
1. Set the graph weights ($K$) as given by (2.2) between each pixel and only its eight nearest neighbors;
2. Derive the Laplacian ($L = D - K$) and perform its eigendecomposition;
3. Compute the GFT of the observed noisy image $\mathbf{y_f}$ as given by (2.4);
4. Perform denoising process by lowpass filtering in the GFT domain as in (2.7);
5. Compute the denoised image $\hat{\mathbf{x}}$ by the inverse GFT as given by (2.5);
```

Table 2.3. *SGM3 Algorithm*

SGM3 :

Step 1

1. Obtain pre-filtered image $\tilde{\mathbf{x}}$ by performing Gaussian filtering;

2. Set the graph weights (K) as given by (2.3) between each pixel and only its eight nearest neighbors;

3. Derive the normalized Laplacian ($\tilde{L} = I - D^{-1/2}KD^{-1/2}$) and perform its eigen-decomposition;

4. Compute the GFT of the observed noisy image $\mathbf{y_f}$ as given by (2.4);

5. Perform denoising process by hard thresholding in the GFT domain as in (2.8);

6. Compute the denoised image $\hat{\mathbf{x}}_\mathbf{1}$ by the inverse GFT as given by (2.5).

Step 2

1. Set the new graph weights (K) from the cleaner output image of the first step $\hat{\mathbf{x}}_\mathbf{1}$ as given by (2.3) with each pixel connected to only its nearest eight neighbors;

2. Derive the normalized Laplacian ($\tilde{L} = I - D^{-1/2}KD^{-1/2}$) and perform its eigen-decomposition;

3. Compute the GFT of the observed noisy image $\mathbf{y_f}$ as given by (2.4) using the new eigenvector matrix V;

4. Perform denoising process by hard thresholding in the GFT domain as in (2.8);

5. Compute the denoised image $\hat{\mathbf{x}}$ by the inverse GFT as given by (2.5) using the new eigenvector matrix V;

---

3. WIENER FILTERING IN THE GRAPH FOURIER DOMAIN

In this chapter, we propose a Wiener filtering scheme implemented directly in GFT domain without requiring power spectral density or correlation function estimation first, to be integrated to various spectral graph based image denoising algorithms as a post-processing step to further improve their already achieved denoising accuracies. This is the main contribution of our thesis.

The Wiener filter coefficients in the GFT domain, that minimize the MSE

$$\mathrm{MSE} = E\left\{\|\mathbf{x} - \hat{\mathbf{x}}\|^2/N\right\}$$

between original, noiseless and denoised images, are derived in [6, 25–29] as

$$W(\lambda_l) = \frac{x_f^2(\lambda_l)}{x_f^2(\lambda_l) + \sigma^2} \tag{3.1}$$

with $x_f(\lambda_l)$, $l = 0, 1, \ldots, N-1$ denoting GFT coefficients of the noiseless image. $E\{\cdot\}$ denotes the statistical expectation operation above. This form is valid for a real, orthonormal V matrix. This true Wiener filter requires unknown GFT coefficients of the original, noiseless image. In [29], an estimated MSE is optimized with respect to truncation of filter eigenmodes and iteration number while iteratively applying the denoising filter, that yields a restricted optimization of filter eigenvalues instead of a true Wiener filtering; whereas, in this work, we attempt to approach the optimal filter eigenvalues given in (9) based on knowledge of the observed noisy image, only, to minimize the MSE.

To approximate the true Wiener filter in (9), GFT coefficients of the noiseless image and the noise standard deviation σ should be estimated. Squared GFT coefficients of the noiseless image are estimated by using processed GFT coefficients of the noisy image, i.e., GFT coefficients of the denoised image obtained by any method described in the previous section, and the estimated noise variance, as

$$\widehat{x_f^2}(\lambda_l) = \hat{x}_f^2(\lambda_l) - \hat{\sigma}^2, \tag{3.2}$$

for $l = 0, 1, \ldots, N-1$, in the form suggested for discrete Fourier, cosine and wavelet transform domains in [7, 30]. $\hat{\sigma}$ denotes the estimated noise standard deviation, computed from the noisy observed 2-D image Y by the methods proposed in [31].

While [7, 30] use transform coefficients of the input noisy image in the above form, we employ GFT coefficients of the denoised image above, in this work, to obtain much better and stabilized results.

Substituting (10) into (9), we obtain the proposed empirical Wiener frequency response as

$$\hat{W}(\lambda_l) = \max\left\{0, \frac{\hat{x}_f^2(\lambda_l) - \hat{\sigma}^2}{\hat{x}_f^2(\lambda_l)}\right\}, \tag{3.3}$$

for $l = 0, 1, \ldots, N-1$.

The proposed Wiener filter can be applied either to the observed, noisy image $\mathbf{y}$ as

$$\hat{x}_{W,f}(\lambda_l) = \hat{W}(\lambda_l) y_f(\lambda_l) \tag{3.4}$$

or to the denoised image $\hat{\mathbf{x}}$ itself obtained by any spectral graph based method to which it is appended, as

$$\hat{x}_{W,f}(\lambda_l) = \hat{W}(\lambda_l) \hat{x}_f(\lambda_l), \tag{3.5}$$

where $\hat{x}_{W,f}(\lambda_l)$, $l = 0, 1, \ldots, N-1$ denote GFT coefficients of the output image of the Wiener filter, $\hat{\mathbf{x}}_{\mathbf{W}}$. This Wiener filter output is finally computed by the inverse GFT,

$$\hat{\mathbf{x}}_{\mathbf{W}} = V \hat{\mathbf{x}}_{\mathbf{W,f}}$$

Our simulations indicate that applying the Wiener filter to the denoised image gives better results. Therefore, we adopt the second approach constituted by (3.3) and (3.5) as our graph Wiener filtering scheme. We append this Wiener filter to the algorithms presented in the previous section as a post-processing step to improve their denoising accuracies. Integrated algorithms are designated as SGM1W, SGM2W and SGM3W.

4. EXPERIMENTAL RESULTS

We apply three spectral graph based algorithms presented in Chapter II, their improved versions by the proposed Wiener filter in Chapter III, and the BM3D algorithm for denoising 256 × 256 benchmark gray scale images (available at $http : //www.cs.tut.fi/{\sim}foi/GCF - BM3D/BM3D\_images.zip$). Noise standard deviation is first taken as $\sigma = 15$ for additive, white, zero-mean Gaussian noise. We compare PSNR values in dB ($10\log(255^2/\text{MSE})$) averaged over 20 independent realizations given by these methods in Table 4.1. Except for the stripes image, spectral graph based methods and Wiener filters appended to them are applied to each image patch of size 64 × 64 in a locally adaptive manner, as also proposed for local adaptive Wiener filters in fixed transform domains [30].

Table 4.1. *PSNR values (in dB) for SGM1, SGM1W, SGM2, SGM2W, SGM3, SGM3W and BM3D algorithms.*

	Box	Stripes	House	C.man	Peppers	Montage
SGM1	47.409	55.279	31.744	29.897	30.403	32.641
SGM1W	**53.105**	**58.017**	32.658	30.281	31.232	33.834
SGM2	48.225	45.683	32.099	29.626	30.754	32.050
SGM2W	**53.051**	45.772	32.884	30.216	31.293	33.564
SGM3	57.249	45.733	30.022	27.672	27.193	29.729
SGM3W	**57.269**	45.734	30.152	28.324	27.447	30.257
BM3D	45.859	46.587	34.932	31.585	32.687	35.097

In Table 4.1, denoising accuracies of considered spectral graph based algorithms, their versions improved by the proposed Wiener filter and the BM3D algorithm are presented for some standard benchmark images. Spectral graph based algorithms and their versions integrated with the Wiener filter have significantly surpassed the BM3D algorithm for the simpler box image. SGM1 and SGM1W have outstanding performances for the stripes image as compared to BM3D. This is also the case for SGM3 and SGM3W for the box image. For other four images that are more complicated, the BM3D has performed better. However, for all images, results obtained by spectral graph algorithms have been consistently improved by the Wiener filter post-processing, especially for SGM1W and SGM2W over SGM1 and SGM2, respectively.

Table 4.2. *PSNR values (in dB) for the stripes image for SGM1, SGM1W and BM3D algorithms for different noise standard deviation values.*

	$\sigma = 5$	$\sigma = 10$	$\sigma = 15$	$\sigma = 20$	$\sigma = 25$
SGM1	65.081	59.749	55.279	53.206	41.214
SGM1W	**67.577**	**62.484**	**58.017**	**54.859**	**43.013**
BM3D	56.141	50.123	46.587	44.102	42.144

In Table 4.2, PSNR values for SGM1 and its improved version with the Wiener filter, SGM1W, are presented to compare with the BM3D, for the stripes image only, for different noise standard deviations. Proposed combined algorithm SGM1W has a superiority to BM3D for all values of standard deviations. Especially for lower ones, obtained denoising accuracies are extremely remarkable. However, for $\sigma = 25$, there is a clear deterioration in the result given by SGM1; it is surpassed by the BM3D. For this case, the proposed Wiener filter step enabled the combined spectral graph method SGM1W to surpass the BM3D. This deterioration hints that eigendecomposition employed in spectral graph methods becomes less stable with increasing noise levels.

The proposed Wiener filter post-processing has improved performances of three spectral graph methods considered, in all cases. Improvements are significant for simpler box and stripes images. The improvement is not quite significant for SGM3 that employs hard thresholding in the GFT domain, than SGM1 and SGM2. This is because Wiener filter does not change nulled GFT coefficients.

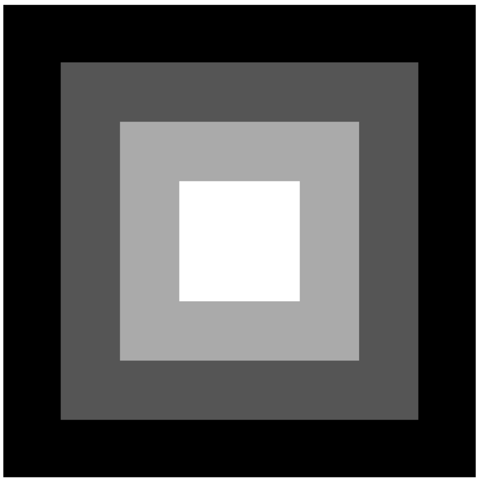

Figure 4.1. *Original box image.*

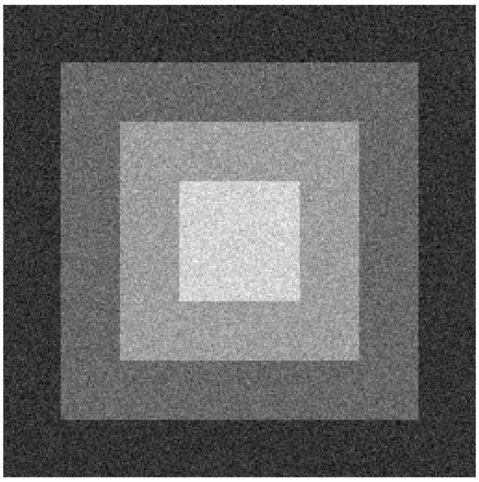

Figure 4.2. *Noisy box image.*

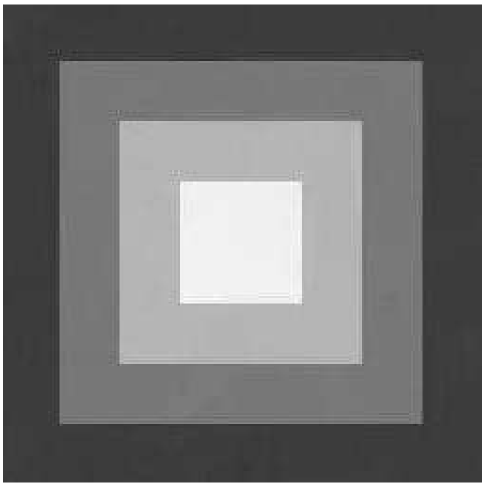

Figure 4.3. *Denoised box image by the SGM3 algorithm.*

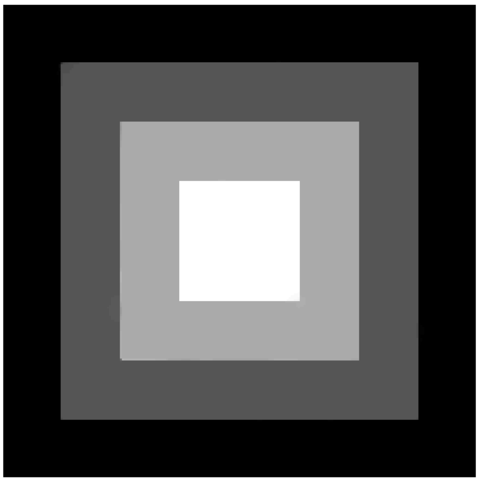

Figure 4.4. *Denoised box image by the SGM3W algorithm.*

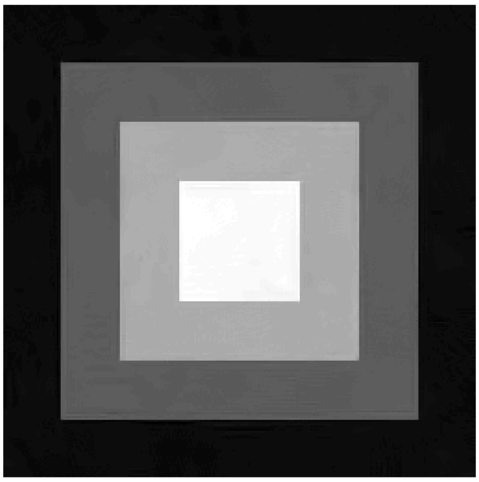

Figure 4.5. *Denoised box image by the BM3D algorithm.*

Figure 4.1 and Figure 4.2 present noiseless and noisy (with $\sigma = 15$) box images, respectively. Figure 4.3 displays denoised image obtained by the SGM3 algorithm. Improved denoising by the SGM3W algorithm employing the Wiener filter as post-processing is illustrated in Figure 4.4. Finally, denoised version of the box image obtained by the BM3D algorithm is given in Figure 4.5. It can be easily seen that the proposed algorithm is quite effective in deblurring the original image edges and attaining the true pixel values of the original noiseless image. Furthermore, SGM3W is seen to recover flat regions better than the BM3D algorithm for this case. Its output image is almost indistinguishable from the noiseless image.

Table 4.3. *PSNR values (in dB) for different regions of the box image for SGM1, SGM1W algorithms.*

	Flat Region	*Edge Region*	*Corner Region*
SGM1	65.081	59.749	55.279
SGM1W	67.577	62.484	58.017

In Table 4.3, PSNR values for SGM1 and its improved version with the Wiener filter, SGM1W, are presented for different regions of the box image. These algorithms are applied to each region of a patch of size 15×15 in a locally adaptive manner. Obtained results demonstrate that spectral graph based algorithm and its improved version with the Wiener filter perform denoising better on flat regions as compared to edge and corner regions of the image.

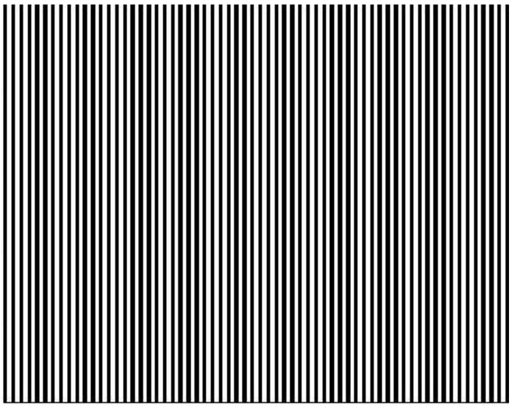

Figure 4.6. *Original stripes image.*

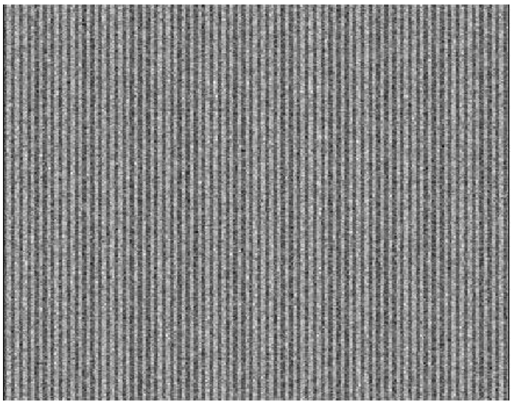

Figure 4.7. *Noisy stripes image.*

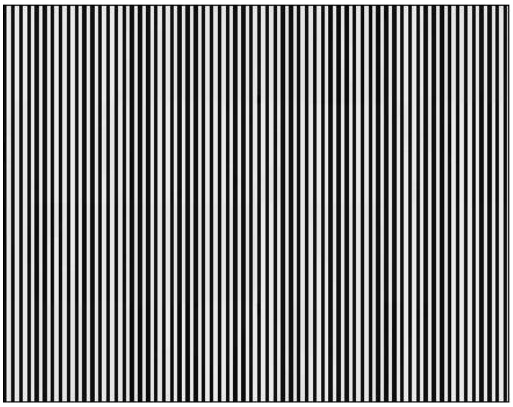

Figure 4.8. *Denoised stripes image by the SGM1 algorithm.*

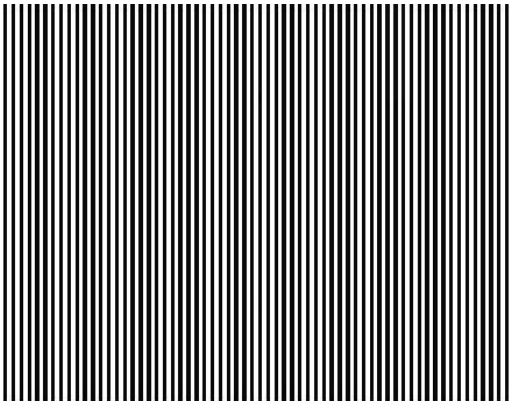

Figure 4.9. *Denoised stripes image by the SGM1W algorithm.*

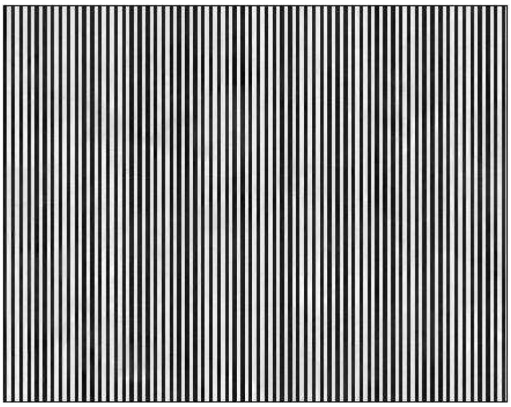

Figure 4.10. *Denoised stripes image by the BM3D algorithm.*

For the stripes image, original noiseless and noisy (with $\sigma = 15$) images are displayed in Figure 4.6 and Figure 4.7, respectively. Figure 4.8 presents denoised image obtained by the SGM1 algorithm. Denoised image by the proposed algorithm SGM1W (with the Wiener filter), is given in Figure 4.9. Finally, the denoised image given by BM3D is displayed in Figure 4.10. Denoised images obtained by these three algorithms can not be distinguished from each other, due to the texture of the stripes image. However, it has been shown in Table 4.2 that PSNR values achieved by the SGM1W algorithm are significantly higher than those of the BM3D algorithm, for all standard deviation values from 5 to 25.

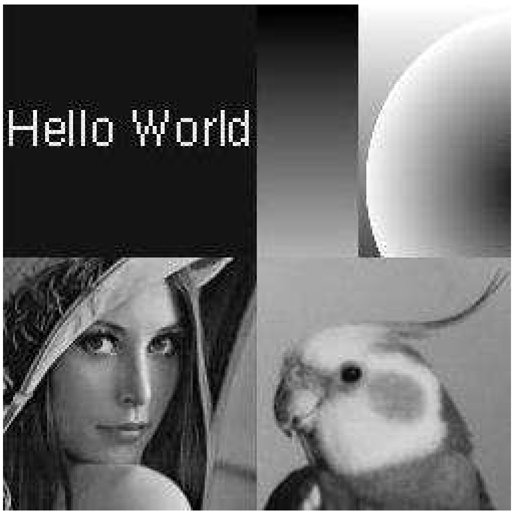

Figure 4.11. *Original montage image.*

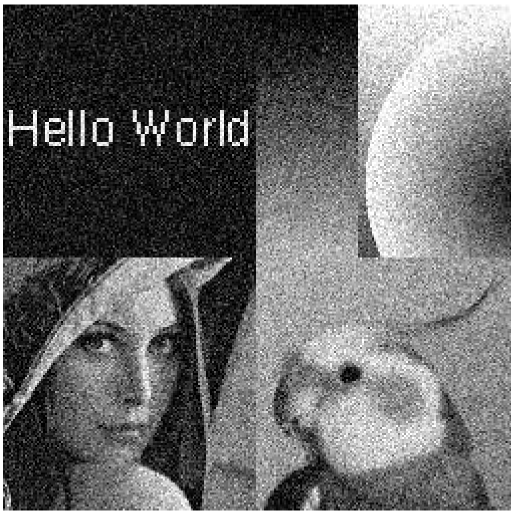

Figure 4.12. *Noisy montage image.*

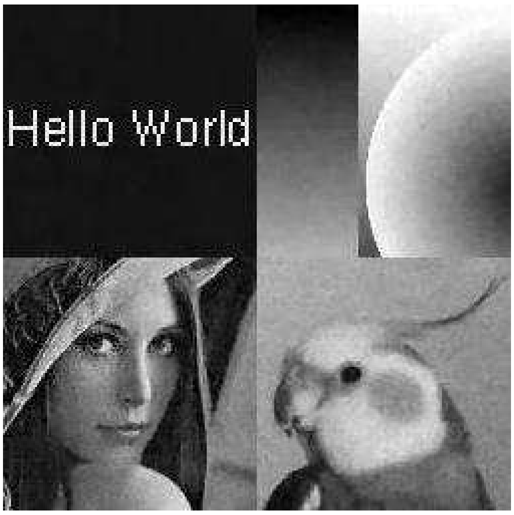

Figure 4.13. *Denoised montage image by the SGM1 algorithm.*

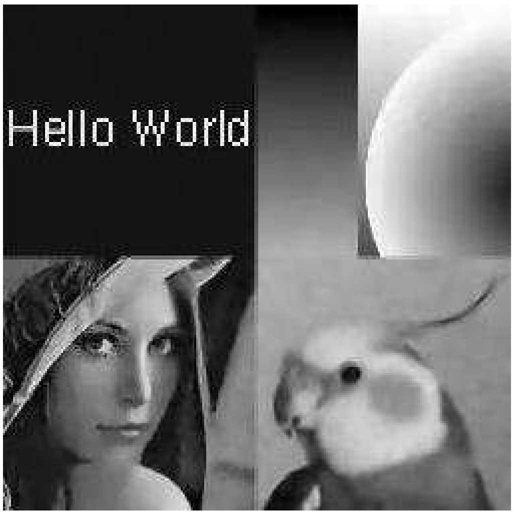

Figure 4.14. *Denoised montage image by the SGM1W algorithm.*

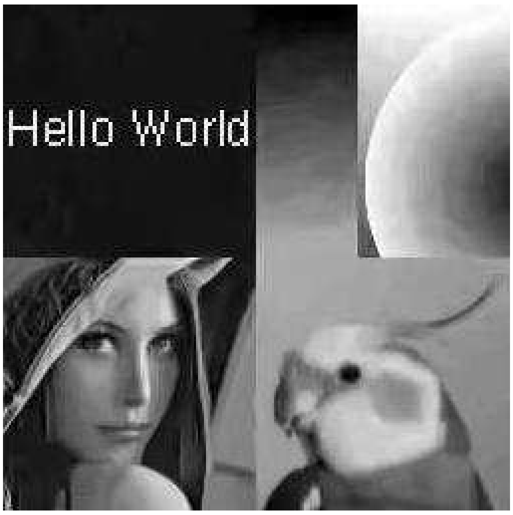

Figure 4.15. *Denoised montage image by the BM3D algorithm.*

Figure 4.11 and Figure 4.12 display original noiseless and noisy (with $\sigma = 15$) images, respectively. Denoised image obtained by the SGM1 algorithm is given in Figure 4.13. Denoised images obtained by SGM1W and BM3D algorithms are presented in Figure 4.14 and Figure 4.15, respectively. The montage image has four different subimages; two upper subimages are less textured, simpler images, and the lower ones are more complicated benchmark images. Therefore, our proposed algorithm SGM1W could surpass the BM3D algorithm for the two upper simpler subimages only.

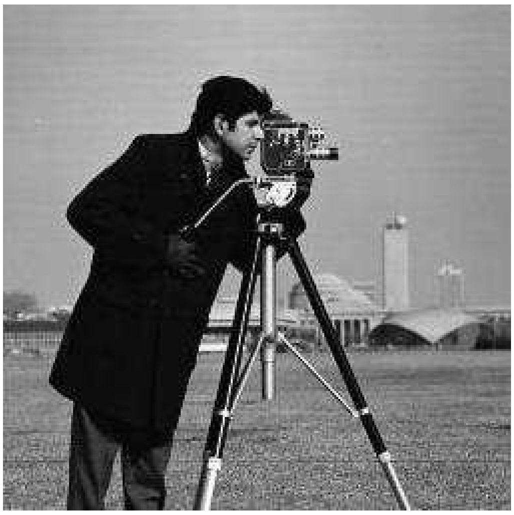

Figure 4.16. *Original cameraman image.*

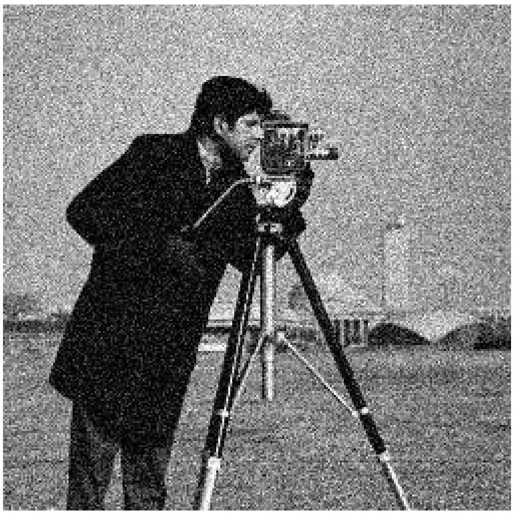

Figure 4.17. *Noisy cameraman image.*

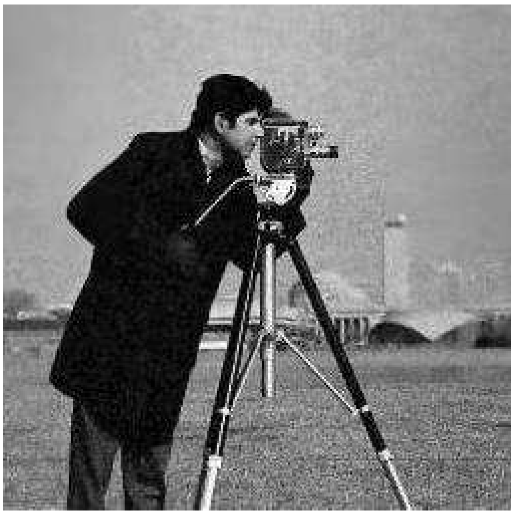

Figure 4.18. *Denoised cameraman image by the SGM1 algorithm.*

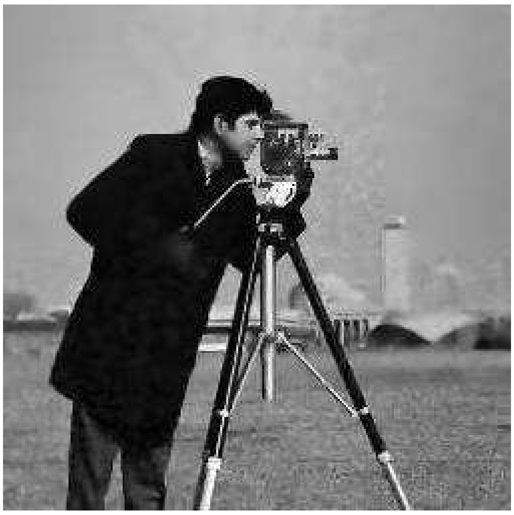

Figure 4.19. *Denoised cameraman image by the SGM1W algorithm.*

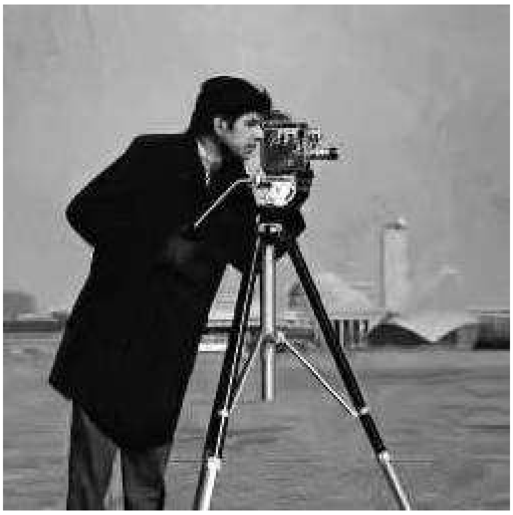

Figure 4.20. *Denoised cameraman image by the BM3D algorithm.*

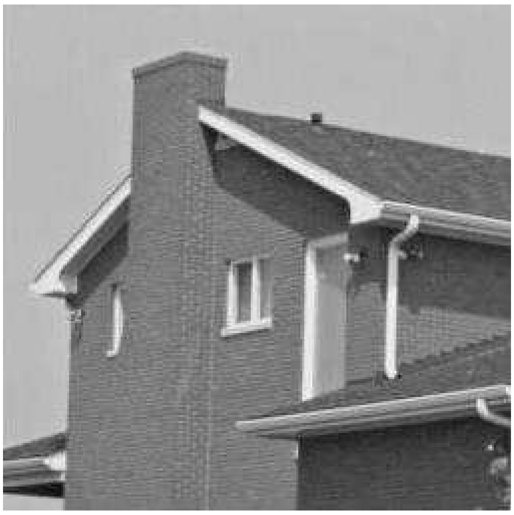

Figure 4.21. *Original house image.*

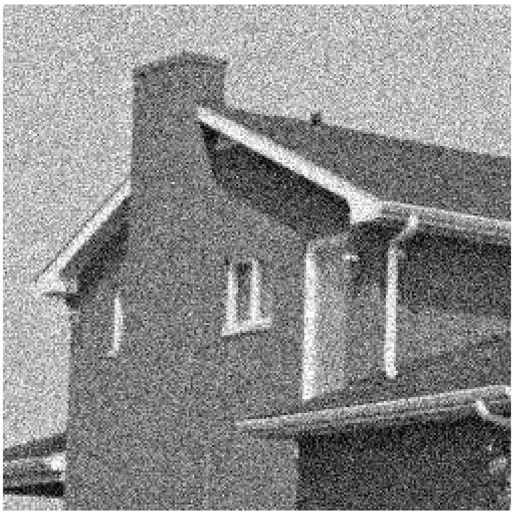

Figure 4.22. *Noisy house image.*

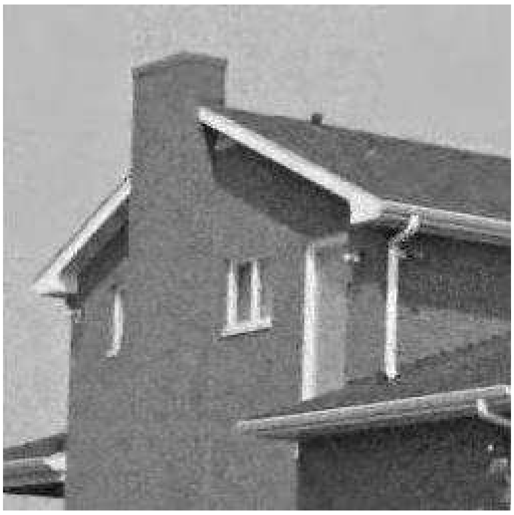

Figure 4.23. *Denoised house image by the SGM1 algorithm.*

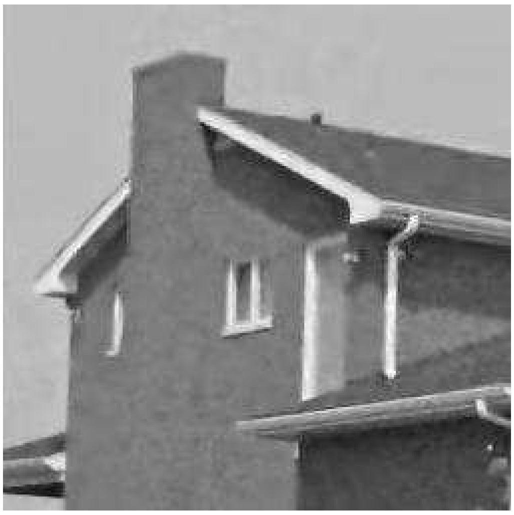

Figure 4.24. *Denoised house image by the SGM1W algorithm.*

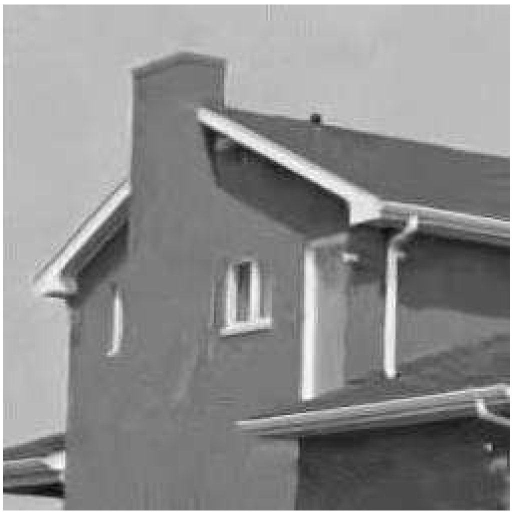

Figure 4.25. *Denoised house image by the BM3D algorithm.*

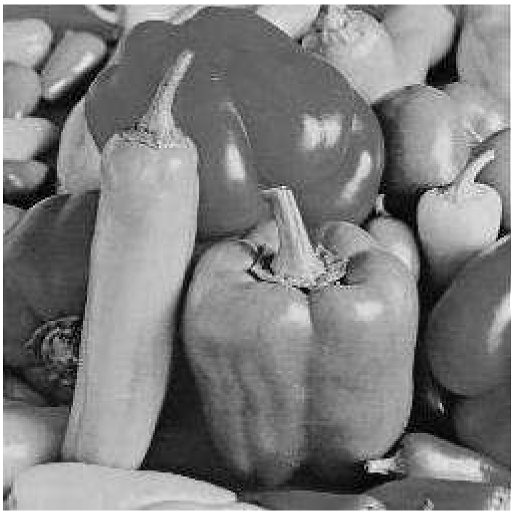

Figure 4.26. *Original peppers image.*

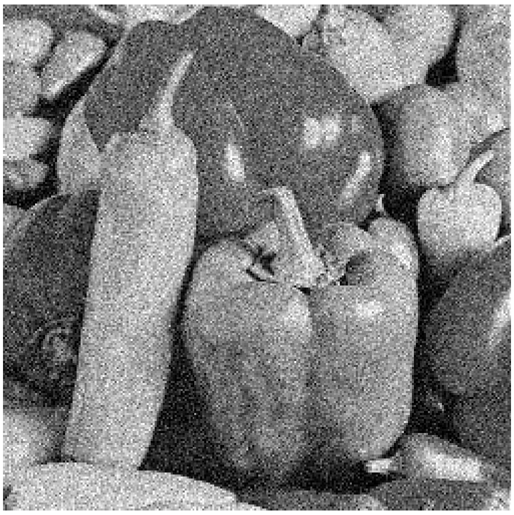

Figure 4.27. *Noisy peppers image.*

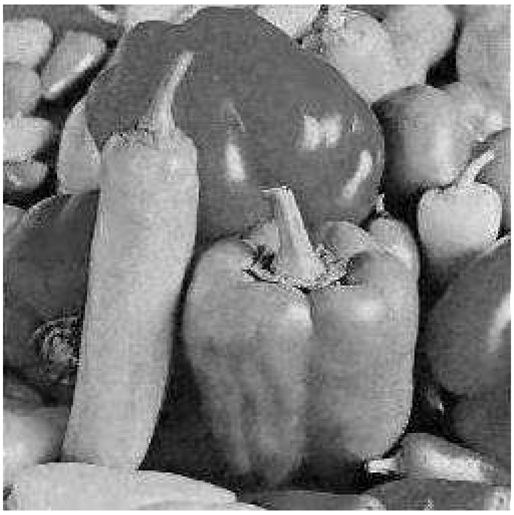

Figure 4.28. *Denoised peppers image by the SGM1 algorithm.*

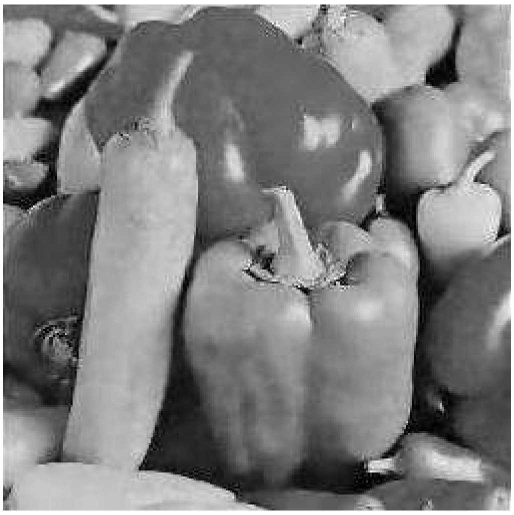

Figure 4.29. *Denoised peppers image by the SGM1W algorithm.*

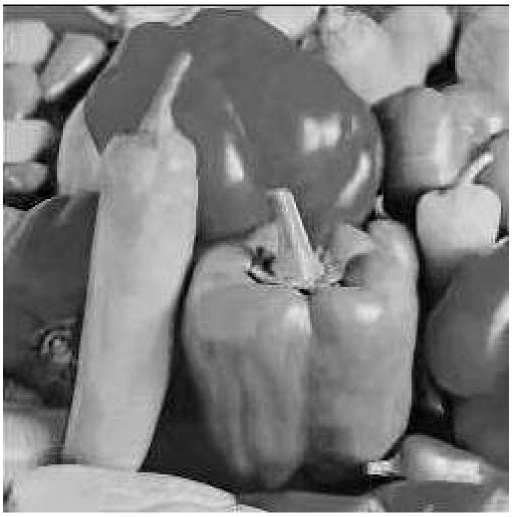

Figure 4.30. *Denoised peppers image by the BM3D algorithm.*

Original, noisy and denoised versions of the other three benchmark images, i.e., cameraman, house and peppers images are presented in the rest of the figures. Since these images are comparatively more textured and complicated, the BM3D has performed better than the spectral graph based methods. A clear improvement in visual quality is observed for denoised images obtained by the SGM1W algorithm over those obtained by the SGM1 algorithm. This visual improvement is also in agreement with comparison of the PSNR values of these two methods given in Table 4.1. Since the edges have been preserved on denoised images obtained by the SGM1W algorithm, this proposed algorithm can also compete with the BM3D algorithm in edge recovery.

5. CONCLUSION

We have proposed a Wiener filter in the GFT domain to improve denoising accuracies of spectral graph based image denoising methods. Our results indicate that this Wiener filter applied as a post-processing step yields consistent accuracy improvement for different choices of graph weights and graph Laplacian matrices and for different processing methods used to denoise obtained GFT coefficients. With this improvement, combined spectral graph methods have attained significantly higher PSNR values than the BM3D method, for box and stripes images, and for upper two subimages of the montage image that are simple enough to be modeled as deterministic images. This suggests that proposed approach has the potential of improving denoising accuracy especially for such less textured, smoother, simpler structured images that can be viewed as deterministic, as hinted by bounds derived in [1]. When integrated into the second stage of the two-stage algorithm that surpasses the BM3D algorithm for a set of benchmark images [22], the Wiener filter step is expected to yield even better results, since it improves the results of SGM3 that uses the same type of graph weights.

The proposed combined filtering is usually applied to each image patch adapted to its local structure, rather than to the entire image, for better performance. Therefore, it can be regarded as a simple and decimated form of a space varying or space-frequency (vertex-frequency) filter in a windowed graph Fourier transform domain [32], since the graph shift operation matches the spatial shift operation for a regular graph as an image. Such a filtering is reminiscent to denoising in a spectral graph wavelet domain [33].

REFERENCES

[1] P. Chatterjee and P. Milanfar. Is denoising dead?. *IEEE Transactions on Image Processing*, 19(4):895–911, 2010.

[2] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Proceedings of the 6th IEEE International Conference on Computer Vision.* pages 839–846, Bombay, India, 1998.

[3] H. Knutsson and C. F. Westin. Normalized and differential convolution: Methods for interpolating and filtering of incomplete and uncertain data. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 515–523, 1993.

[4] L. Yaroslavsky. *Digital Picture Processing: Springer-Verlag.* Berlin, Germany, 1987.

[5] S. M. Smith and J. M. Brady. Susan: A new approach to low level image processing. *International Journal of Computer Vision*, 23(1):45–78, 1997.

[6] P. Milanfar. A tour of modern image filtering: New insights and methods, both practical and theoretical. *IEEE Signal Processing Magazine*, 30(1):106–128, 2013.

[7] A. Buades, B. Coll, and J.M. Morel. A review of image denoising algorithms, with a new one. *SIAM Journal on Multiscale Modeling and Simulation: A SIAM Interdisciplinary Journal*, 4(2):490–530, 2005.

[8] S. P. Awate and R. T. Whitaker. Unsupervised, information-theoretic, adaptive image filtering for image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(3): 364–376, 2006.

[9] C. Kervrann and J. Boulanger. Optimal spatial adaptation for patch-based image denoising. *IEEE Transactions on Image Processing*, 15(10):2866–2878, 2006.

[10] C. Kervrann and J. Boulanger. Local adaptivity to variable smoothness for exemplar-based image regularization and representation. *International Journal of Computer Vision*, 79(1):45–69, 2008.

[11] H. Takeda, S. Farsiu, and P. Milanfar. Kernel regression for image processing

and reconstruction. *IEEE Transactions on Image Processing*, 16(2):349–366, 2007.

[12] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(2):3736–3745, 2006.

[13] P. Chatterjee and P. Milanfar. Clustering-based denoising with locally learned dictionaries. *IEEE Transactions on Image Processing*, 18(7):1438–1451, 2009.

[14] P. Chatterjee and P. Milanfar. Patch-based near-optimal image denoising. *IEEE Transactions on Image Processing*, 21(4):1635–1649, 2012.

[15] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, 2007.

[16] F. Zhang and E.R. Hancock. Graph spectral image smoothing using the heat kernel. *Pattern Recognition*, 41(7):3328–3342, 2008.

[17] A. Elmoataz, O. Lezoray, and S. Bougleux. Nonlocal discrete regularization on weighted graphs: A framework for image and manifold processing. *IEEE Transactions on Image Processing*, 17:1047–1060, 2008.

[18] D.I. Shuman, S.K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, 2013.

[19] A. Kheradmand and P. Milanfar. A general framework for kernel similarity-based image denoising. In *Proceedings of IEEE Global Conference on Signal Information Processing (GlobalSIP)*, pages 415–418, 2013.

[20] A. Sandryhaila and J.M.F. Moura. Discrete signal processing on graphs. *IEEE Transactions on Signal Processing*, 61(7):1644–1656, 2013.

[21] S. Chen, A. Sandryhaila, J.M.F. Moura, and J. Kovačević. Signal denoising on graphs via graph filtering. In *Proceedings of IEEE Global Conference on Signal Information Processing (GlobalSIP).* pages 872–876, 2014.

[22] F.G. Meyer and X. Shen. Perturbation of the eigenvectors of the graph laplacian: Application to image denoising. *Applied and Computational Harmonic Analysis*, 36(2):326–334, 2014.

[23] A. Gadde, S.K. Narang, and A. Ortega. Bilateral filter: Graph spectral interpretation and extensions. In *Proceedings of IEEE International Conference on Image Processing (ICIP)*, pages 1222–1226, 2013.

[24] M. Onuki, S. Ono, M. Yamagishi, and Y. Tanaka. Graph signal denoising via trilateral filter on graph spectral domain. *IEEE Transactions on Signal and Information Processing over Networks*, 2(2):137–148, 2016.

[25] B. Girault, P. Gonçalves, E. Fleury, and A.S. Mor. Semi-supervised learning for graph to signal mapping: A graph signal wiener filter interpretation. In *Proceedings of IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP)*, pages 1115–1119, 2014.

[26] N. Perraudin and P. Vandergheynst. Stationary signal processing on graphs. *arXiv preprint arXiv:1601.02522*, 2016.

[27] A. Loukas and D. Foucard. Frequency analysis of temporal graph signals. *arXiv preprint arXiv:1602.04434*, 2016.

[28] A. Gavili and X.-P. Zhang. On the shift operator, graph frequency and optimal filtering in graph signal processing. *arXiv preprint arXiv:1511.03512v3 [math.SP]*, 2016.

[29] H. Talebi and P. Milanfar. Global image denoising. *IEEE Transactions on Image Processing*, 23:755–768, 2014.

[30] L. Yaroslavsky. Nonlinear adaptive image processing in transform domain. In *Proceedings of IEEE Workshop Nonlinear Signal Image Processing*, 1997.

[31] J.Ĩmmerkær. Fast noise variance estimation. *Computer Vision and Image Understanding*, 64(2):300–302, 1996.

[32] D.I. Shuman, B. Ricaud, and P. Vandergheynst. A windowed graph fourier transform. In *Proceedings of IEEE Statistical Signal Processing Workshop*, pages 133–136, 2012.

[33] D.K. Hammond, L. Jacques, and P. Vandergheynst. Image denoising with nonlocal spectral graph wavelets. In *Image Processing and Analysis with Graphs: Theory and Practice, Eds. O. Lézoray and L. Grady.* Chapter 8:208–236, CRC Press, 2012.

APPENDIX

Appendix A: SGM1W Algorithm in MATLAB

SGM1W Algorithm:

Laplacian matrix as Bilateral(Lap) and L_2 regularization.

One-pass denoising.

Laplacian matrix, L_2 regularization in graph FT domain.

$clc; clear \;\; all; close \;\; all;$

$M = 32;$

$N = 32;$

$X = 60 * ones(M, N);$

$X(M/8 : 7 * M/8, N/8 : 7 * N/8) = 120 * ones(3 * M/4 + 1, 3 * N/4 + 1);$

$X(M/4 : 3 * M/4, N/4 : 3 * N/4) = 180 * ones(M/2 + 1, N/2 + 1);$

$X(3 * M/8 : 5 * M/8, 3 * N/8 : 5 * N/8) = 240 * ones(M/4 + 1, N/4 + 1);$

Standard deviation value

$sig = 15;$

$Y = X + sig * randn(M, N);$

$[M, N] = size(Y);$

Window width.

$w = 1;$

Extended input image allocated.

$Yext = zeros(M + w - 1, N + w - 1);$

Adjacency (or weight) matrix based on pixel locations allocated.

$Kx = zeros(M * N, M * N);$

Adjacency (or weight) matrix based on pixel values allocated.

$Ky = zeros(M * N, M * N);$

Initial gaussian filter smoothing parameter.

Gaussian smoothing parameter for the pixel location based weight matrix.

$hx = 1;$

Smoothing parameter for the pixel value based weight matrix.

$hy = 75;$

Noisy image to be denoised vectorized columnwise.

$y = reshape(Y, M * N, 1);$

K and L vectors denoted as column and line indexes, of size $M * N$, respectively.

$[K, L] = meshgrid(1 : M * N, 1 : M * N);$

$Mcol = ceil(K/M);$

$Ncol = ceil(L/M);$

$Mrow = rem(K, M);$

$Nrow = rem(L, M);$

$Mrow = Mrow + M * (Mrow == 0);$

$Nrow = Nrow + M * (Nrow == 0);$

Kx is Gaussian Filter impulse response matrix.

Adjacency (weight) matrix based on Pixel Location.

$Kx = exp(-((Mrow - Nrow).^2 + (Mcol - Ncol).^2)/hx);$

Initial denoising by Gaussian filter.

$xinit = diag(1./sum(Kx.')) * Kx * y;$

Initially denoised image set.

$Xinit = reshape(xinit, M, N);$

Extended image set, symmetrically.

$Yext((w + 1)/2 : M + (w - 1)/2, (w + 1)/2 : N + (w - 1)/2) = Xinit;$

$for\ \ k = 1 : (w - 1)/2$

$Yext((w + 1)/2 - k, :) = Yext((w + 1)/2 + k, :);$

$Yext(M + (w - 1)/2 + k, :) = Yext(M + (w - 1)/2 - k, :);$

$Yext(:, (w + 1)/2 - k) = Yext(:, (w + 1)/2 + k);$

$Yext(:, N + (w - 1)/2 + k) = Yext(:, N + (w - 1)/2 - k);$

$end;$

Adjacency (weight) matrix based on pixel locations (Kx), and, pixel values as in NLM (Ky).

$for\ \ m = 1 : M * N$

$for\ \ n = 1 : M * N$

$Ky(m, n) = exp(-norm(Yext(Mrow(1, m) : Mrow(1, m) + w - 1, Mcol(1, m) :$

$Mcol(1, m) + w - 1) - Yext(Nrow(n, 1) : Nrow(n, 1) + w - 1, Ncol(n, 1) :$

$Ncol(n, 1) + w - 1), "fro")^2/hy);$

$end;$

$end;$

New smoothing parameter Guassian.

$hx = 0.5;$

Adjacency (weight) matrix based on pixel locations.

$Kx = exp(-((Mrow - Nrow).^2 + (Mcol - Ncol).^2)/hx);$

Kernel matrix as in the bilateral filter.

$K = Kx. * Ky;$

Graph Laplacian matrix based on the weight matrix of the Bilateral filter.

$Lap = diag(sum(K.')) - K;$

Eigendecomposition of Laplacian matrix.

$[V, D] = eig(Lap);$

Graph FT of the input noisy image to be denoised.

$yf = V' * y;$

Parameter of low pass filter.

$alpha = 250;$

L2 regularization (low pass filtering process).

$yf = (1./(1 + alpha * diag(D))). * yf;$

Inverse graph FT of thresholded transform to give denoised image vector.

$xdenoised = V * yf;$

Denoised image set.

$Xdenoised = reshape(xdenoised, M, N);$

MSE between original and denoised image.

$MSE = mean(mean((X - Xdenoised).^2))$

Plotting the original image.

$figure; colormap(gray); pcolor(X); shadingflat; title('Original\ \ image');$

Plotting the denoised image with SGM1 algorithm.

$figure; colormap(gray); pcolor(Xdenoised); shadingflat; title('Denoised\ \ image');$

Proposed Graph Wiener filter for improved performance:

Estimation of noise standart deviation

$H = [1\ \ -2\ \ 1; -2\ \ 4\ \ -2; 1\ \ -2\ \ 1];$

Mask filter

$Z = conv2(Y, H,' valid');$

$sigest = mean(mean(abs(Z)))/6 * sqrt(pi/2);$

Estimated frequency response of the Wiener filter from graph FT of the denoised image.

$wien = max(0, (yf.^2 - sigest^2)./yf.^2);$

Output of the Wiener filter applied to the denoised image.

$xwien = V * (wien. * yf);$

$Xwien = reshape(xwien, M, N);$

MSE between original and Wiener denoised image.

$MSEW = mean(mean((X - Xwien).^2))$

Plotting the denoised image with SGM1W algorithm (the improved version with the Wiener filter).

$figure; colormap(gray); pcolor(Xwien); shadingflat; title('Wiener\ \ Filtered\ \ image');$

Appendix B: SGM2W Algorithm in MATLAB

SGM2W Algorithm:

Laplacian matrix as Bilateral(Lap) and L_2 regularization

Image pixels are connected to their 8 nearest neighbors.

One-pass denoising.

Laplacian matrix, L_2 regularization in graph FT domain.

$clc; clear\ \ all; close\ \ all;$

$M = 32;$

$N = 32;$

$X = 60 * ones(M, N);$

$X(M/8 : 7 * M/8, N/8 : 7 * N/8) = 120 * ones(3 * M/4 + 1, 3 * N/4 + 1);$

$X(M/4 : 3 * M/4, N/4 : 3 * N/4) = 180 * ones(M/2 + 1, N/2 + 1);$

$X(3 * M/8 : 5 * M/8, 3 * N/8 : 5 * N/8) = 240 * ones(M/4 + 1, N/4 + 1);$

Standard deviation value.

$sig = 15;$

$Y = X + sig * randn(M, N);$

$[M, N] = size(Y);$

Window width.

$w = 1;$

Extended input image allocated.

$Yext = zeros(M + w - 1, N + w - 1);$

Adjacency (or weight) matrix based on pixel locations allocated.

$Kx = zeros(M * N, M * N);$

Adjacency (or weight) matrix based on pixel values allocated.

$Ky = zeros(M * N, M * N);$

Initial gaussian filter smoothing parameter.

Gaussian smoothing parameter for the pixel location based weight matrix.

$hx = 1;$

Smoothing parameter for the pixel value based weight matrix.

$hy = 75;$

Noisy image to be denoised vectorized columnwise.

$y = reshape(Y, M * N, 1);$

K and L vectors denoted as column and line indexes, of size $M * N$, respectively.

$[K, L] = meshgrid(1 : M * N, 1 : M * N);$

$Mcol = ceil(K/M);$

$Ncol = ceil(L/M);$

$Mrow = rem(K, M);$

$Nrow = rem(L, M);$

$Mrow = Mrow + M * (Mrow == 0);$

$Nrow = Nrow + M * (Nrow == 0);$

Kx is Gaussian Filter impulse response matrisi.

Adjacency (weight) matrix based on Pixel Location.

$Kx = exp(-((Mrow - Nrow).^2 + (Mcol - Ncol).^2)/hx);$

Initial denoising by Gaussian filter.

$xinit = diag(1./sum(Kx.')) * Kx * y;$

Initially denoised image set.

$Xinit = reshape(xinit, M, N);$

Extended image set, symmetrically.

$Yext((w + 1)/2 : M + (w - 1)/2, (w + 1)/2 : N + (w - 1)/2) = Xinit;$

$for\ \ k = 1 : (w - 1)/2$

$Yext((w + 1)/2 - k, :) = Yext((w + 1)/2 + k, :);$

$Yext(M + (w - 1)/2 + k, :) = Yext(M + (w - 1)/2 - k, :);$

$Yext(:, (w + 1)/2 - k) = Yext(:, (w + 1)/2 + k);$

$Yext(:, N + (w - 1)/2 + k) = Yext(:, N + (w - 1)/2 - k);$

$end;$

Adjacency (weight) matrix based on pixel locations (Kx), and, pixel values as in NLM (Ky).

$for\ \ m = 1 : M * N$

$for\ \ n = 1 : M * N$

if

$norm([Mrow(1, m), Mcol(1, m)] - [Nrow(n, 1), Ncol(n, 1)], inf) <= 1$

$Ky(m, n) = exp(-norm(Yext(Mrow(1, m) : Mrow(1, m) + w - 1, Mcol(1, m) :$
$Mcol(1, m) + w - 1) - Yext(Nrow(n, 1) : Nrow(n, 1) + w - 1, Ncol(n, 1) :$
$Ncol(n, 1) + w - 1),' fro')^2/hy);$

$end;$

$end;$

$end;$

$for\ \ m = 1 : M * N$

$for\ \ n = 1 : M * N$

$Ky(m, n) = exp(-norm(Yext(Mrow(1, m) : Mrow(1, m) + w - 1, Mcol(1, m) : Mcol(1, m) + w - 1) - Yext(Nrow(n, 1) : Nrow(n, 1) + w - 1, Ncol(n, 1) : Ncol(n, 1) + w - 1),' fro')^2/hy);$

$end;$

$end;$

New smoothing parameter Guassian.

$hx = 0.5;$

Adjacency (weight) matrix based on pixel locations.

$Kx = exp(-((Mrow - Nrow).^2 + (Mcol - Ncol).^2)/hx);$

Kernel matrix as in the bilateral filter.

$K = Kx. * Ky;$

Graph Laplacian matrix based on the weight matrix of the Bilateral filter.

$Lap = diag(sum(K.')) - K;$

Eigendecomposition of Laplacian matrix.

$[V, D] = eig(Lap);$

Graph FT of the input noisy image to be denoised.

$yf = V' * y;$

Parameter of low pass filter.

$alpha = 250;$

L2 regularization (low pass filtering process).

$yf = (1./(1 + alpha * diag(D))). * yf;$

Inverse graph FT of thresholded transform to give denoised image vector.

$xdenoised = V * yf;$

Denoised image set.

$Xdenoised = reshape(xdenoised, M, N);$

MSE between original and denoised image.

$MSE = mean(mean((X - Xdenoised).^2))$

Plotting the original image.

$figure; colormap(gray); pcolor(X); shadingflat; title('Original\ \ image');$

Plotting the denoised image with SGM1 algorithm.

$figure; colormap(gray); pcolor(Xdenoised); shadingflat; title('Denoised\ \ image');$

Proposed Graph Wiener filter for improved performance:

Estimation of noise standart deviation.

$H = [1\ \ -2\ \ 1; -2\ \ 4\ \ -2; 1\ \ -2\ \ 1];$

Mask filter.

$Z = conv2(Y, H,' valid');$

Estimated noise standard deviation.

$sigest = mean(mean(abs(Z)))/6 * sqrt(pi/2);$

Estimated freq. resp. of the Wiener filter from graph FT of the denoised image.

$wien = max(0, (yf.^2 - sigest^2)./yf.^2);$

Output of the Wiener filter applied to the denoised image.

$xwien = V * (wien. * yf);$

$Xwien = reshape(xwien, M, N);$

MSE between original and Wiener denoised image.

$MSEW = mean(mean((X - Xwien).^2))$

Plotting the denoised image with SGM1W algorithm (the improved version with the Wiener filter).

$figure; colormap(gray); pcolor(Xwien); shadingflat; title('Wiener\ \ Filtered\ \ image');$

Appendix C: SGM3W Algorithm in MATLAB

SGM3W Algorithm:

Graph denoising methods based on graph Laplacian matrix.

Image pixels are connected to their 8 nearest neighbors.

Weights are set as done in [Meyer,Shen].

Two-pass denoising.

Normalized Laplacian matrix, hard-thresholding in graph FT domain.

$clc; clear\ \ all; close\ \ all;$

$M = 64;$

$N = 64;$

$X = 60 * ones(M, N);$

$X(M/8 : 7 * M/8, N/8 : 7 * N/8) = 120 * ones(3 * M/4 + 1, 3 * N/4 + 1);$

$X(M/4 : 3 * M/4, N/4 : 3 * N/4) = 180 * ones(M/2 + 1, N/2 + 1);$

$X(3 * M/8 : 5 * M/8, 3 * N/8 : 5 * N/8) = 240 * ones(M/4 + 1, N/4 + 1);$

Standard deviation value.

$sig = 15;$

$Y = X + sig * randn(M, N);$

$[M, N] = size(Y);$

Window width.

$w = 1;$

Extended input image allocated.

$Yext = zeros(M + w - 1, N + w - 1);$

Parameter that controls the effect of patch similarity.

$beta = 12;$

Parameter that controls the scaling of the overall similarity metric.

$delta = 5;$

Noisy image to be denoised vectorized columnwise.

$y = Y(:);$

K and L vectors denoted as column and line indexes, of size $M * N$, respectively.

$[K, L] = meshgrid(1 : M * N, 1 : M * N);$

$Mcol = ceil(K/M);$

$Ncol = ceil(L/M);$

$Mrow = rem(K, M);$

$Nrow = rem(L, M);$

$Mrow = Mrow + M \ast (Mrow == 0);$

$Nrow = Nrow + M \ast (Nrow == 0);$

Smoothing parameter for the Gaussian prefilter.

$hx = 1;$

Kernel matrix for the Gaussian prefilter.

$Kx = exp(-((Mrow - Nrow).^2 + (Mcol - Ncol).^2)/hx);$

Adjacency (or weight) matrix allocated.

$K = zeros(M \ast N, M \ast N);$

Initial denoising by Gaussian filter.

$xinit = diag(1./sum(Kx.')) \ast Kx \ast y;$

Initially denoised image set.

$Xinit = reshape(xinit, M, N);$

$Yext((w+1)/2 : M + (w-1)/2, (w+1)/2 : N + (w-1)/2) = Xinit;$

Extended image set, symmetrically.

$for\ \ k = 1 : (w-1)/2$

$Yext((w+1)/2 - k, :) = Yext((w+1)/2 + k, :);$

$Yext(M + (w-1)/2 + k, :) = Yext(M + (w-1)/2 - k, :);$

$Yext(:, (w+1)/2 - k) = Yext(:, (w+1)/2 + k);$

$Yext(:, N + (w-1)/2 + k) = Yext(:, N + (w-1)/2 - k);$

$end;$

Adjacency (weight) matrix set.

$for\ \ m = 1 : M \ast N$

$for\ \ n = 1 : M \ast N$

$ifnorm([Mrow(1,m), Mcol(1,m)] - [Nrow(n,1), Ncol(n,1)], inf) <= 1$

$d = norm(Yext(Mrow(1,m) : Mrow(1,m) + w - 1, Mcol(1,m) : Mcol(1,m) + w - 1) - Yext(Nrow(n,1) : Nrow(n,1) + w - 1, Ncol(n,1) : Ncol(n,1) + w - 1), 'fro') + beta \ast norm([Mrow(1,m), Mcol(1,m)] - [Nrow(n,1), Ncol(n,1)]);$

$K(m,n) = exp(-d^2/delta^2);$

$end;$

$end;$

$end;$

Normalized graph Laplacian matrix.

$Lapnor = eye(size(K)) - diag(1./sqrt(sum(K.'))) * K * diag(1./sqrt(sum(K.')));$

$Lapnor = (Lapnor + Lapnor.')/2;$

Eigendecomposition of Laplacian matrix.

$[V, D] = eig(Lapnor);$

Graph FT of the input noisy image to be denoised.

$yf = V' * y;$

Parameter for hard-thresholding.

$threshold = 105;$

Graph FT coefficients of the noisy image are hard-thresholded.

$yf = (abs(yf) >= threshold). * yf;$

Number of kept transform coefficients.

$count_c = sum((abs(yf) >= threshold));$

Inverse graph FT of thresholded transform to give denoised image vector.

$xdenoised = V * yf;$

Denoised image set.

$Xdenoised = reshape(xdenoised, M, N);$

Second pass:

Window width.

$w = 1;$

Extended input image allocated.

$Yext = zeros(M + w - 1, N + w - 1);$

Parameter that controls the effect of patch similarity.

$beta = 15;$

Parameter that controls the scaling of the overall similarity metric.

$delta = 5;$

Noisy image to be denoised vectorized columnwise.

$y = Y(:);$

K and L vectors denoted as column and line indexes, of size $M * N$, respectively.

$[K, L] = meshgrid(1 : M * N, 1 : M * N);$

$Mcol = ceil(K/M);$

$Ncol = ceil(L/M);$

$Mrow = rem(K, M);$

$Nrow = rem(L, M);$

$Mrow = Mrow + M * (Mrow == 0);$

$Nrow = Nrow + M * (Nrow == 0);$

Smoothing parameter for the Gaussian prefilter.

$hx = 1;$

Kernel matrix for the Gaussian prefilter.

$Kx = exp(-((Mrow - Nrow).^2 + (Mcol - Ncol).^2)/hx);$

Adjacency (or weight) matrix allocated.

$K = zeros(M * N, M * N);$

Initial denoising by Gaussian filter.

$xinit = diag(1./sum(Kx.')) * Kx * y;$

Initially denoised image set.

$Xinit = reshape(xinit, M, N);$

$Yext((w + 1)/2 : M + (w - 1)/2, (w + 1)/2 : N + (w - 1)/2) = Xinit;$

Extended image set, symmetrically.

$for\ \ k = 1 : (w - 1)/2$

$Yext((w + 1)/2 - k, :) = Yext((w + 1)/2 + k, :);$

$Yext(M + (w - 1)/2 + k, :) = Yext(M + (w - 1)/2 - k, :);$

$Yext(:, (w + 1)/2 - k) = Yext(:, (w + 1)/2 + k);$

$Yext(:, N + (w - 1)/2 + k) = Yext(:, N + (w - 1)/2 - k);$

$end;$

Adjacency (weight) matrix set.

$for\ \ m = 1 : M * N$

$for\ \ n = 1 : M * N$

$ifnorm([Mrow(1, m), Mcol(1, m)] - [Nrow(n, 1), Ncol(n, 1)], inf) <= 1$

$d = norm(Yext(Mrow(1, m) : Mrow(1, m) + w - 1, Mcol(1, m) : Mcol(1, m) + w - 1) - Yext(Nrow(n, 1) : Nrow(n, 1) + w - 1, Ncol(n, 1) : Ncol(n, 1) + w - 1), ' fro') + beta * norm([Mrow(1, m), Mcol(1, m)] - [Nrow(n, 1), Ncol(n, 1)]);$

$K(m, n) = exp(-d^2/delta^2);$

$end;$

$end;$

$end;$

Normalized graph Laplacian matrix.

$Lapnor = eye(size(K)) - diag(1./sqrt(sum(K.'))) * K * diag(1./sqrt(sum(K.')));$

$Lapnor = (Lapnor + Lapnor.')/2;$

Eigendecomposition of Laplacian matrix.

$[V, D] = eig(Lapnor);$

Graph FT of the input noisy image to be denoised.

$yf = V' * y;$

Parameter for hard-thresholding

$threshold = 105;$

Graph FT coefficients of the noisy image are hard-thresholded.

$yf = (abs(yf) >= threshold). * yf;$

Number of kept transform coefficients.

$count_c = sum((abs(yf) >= threshold));$

Inverse graph FT of thresholded transform to give denoised image vector.

$xdenoised = V * yf;$

Denoised image set.

$Xdenoised = reshape(xdenoised, M, N);$

MSE between original and denoised image.

$MSE = mean(mean((X - Xdenoised).^2))$

Plotting the original image.

$figure; colormap(gray); pcolor(X); shadingflat; title('Original\ \ image');$

Plotting the denoised image with SGM1 algorithm.

$figure; colormap(gray); pcolor(Xdenoised); shadingflat; title('Denoised\ \ image');$

Proposed Graph Wiener filter for improved performance:

Estimation of noise standart deviation.

$H = [1\ \ -2\ \ 1; -2\ \ 4\ \ -2; 1\ \ -2\ \ 1];$ Mask filter.

$Z = conv2(Y, H,' valid');$

Estimated noise standard deviation.

$sigest = mean(mean(abs(Z)))/6 * sqrt(pi/2);$

Estimated freq. resp. of the Wiener filter from graph FT of the denoised image.

$wien = max(0, (yf.^2 - sigest^2)./yf.^2);$

Output of the Wiener filter applied to the denoised image.

$xwien = V * (wien. * yf);$

$Xwien = reshape(xwien, M, N);$

MSE between original and Wiener denoised image.

$MSEW = mean(mean((X - Xwien).^2))$

Plotting the denoised image with SGM1W algorithm (the improved version with the Wiener filter).

$figure; colormap(gray); pcolor(Xwien); shadingflat; title('Wiener\ \ Filtered\ \ image');$

Ali Can YAĞAN

Curriculum Vitae

PERSONAL DETAILS

Birth	March 4, 1991
Address	Engineering Faculty, Anadolu University,Eskişehir
Phone	+90 555 5413182
Mail	`alicanyagan@anadolu.edu.tr`
Website	`http://www.eem.anadolu.edu.tr/alicanyagan`

EDUCATION

MSc. Electrical and Electronics Engineering Program 2014-2016

Anadolu University

Graduated with GPA: 3.79

BSc. Electrical and Electronics Engineering 2008-2013

Mustafa Kemal University

Graduated with GPA: 2.75

WORK EXPERIENCE

Research Assistant 2014-present

Anadolu University

I am working as a research assistant in the Department of Electrical and Electronics Engineering in Anadolu University.

SKILLS

Languages	Turkish (mother tongue) English (very good) German (basic)
Software	MATLAB, SIMULINK, LaTeX, MICROSOFT OFFICE