TÉCNICO LISBOA

Single-Frame Image Denoising with Gaussian Mixtures

Afonso Morato Alface Martins Teodoro

Introduction to the Research and Design in Electrical and Computer Engineering Report

Examination Committee

Chairperson:	Nome do Presidente
Supervisor:	Professor Doutor Mário Alexandre Teles Figueiredo
Co-supervisor:	Doutora Mariana Sá Correia Leite de Almeida
Members of the committee:	Nome do Vogal 1
	Nome do Vogal 2

January 2014

Contents

List of Figures

Chapter 1

Introduction

This report was made within the scope of the Introduction to the Research and Design in Electrical and Computer Engineering course and serves as an introduction to the dissertation to obtain the Master of Science Degree. This course constitutes an opportunity to become familiarized with the necessary methodologies of project and investigation and to start developing relevant work for the thesis, namely bibliography search and state of the art review.

1.1 Motivation

With the technological development, there is an increasingly higher number of applications that, at some point, involve acquiring one or more images, be it for recreational, medical, surveillance, astronomical, or any one of many other purposes. Naturally, the development of methods to improve the quality of these images go hand-in-hand.

Image denoising is a well-known problem of image processing; it was one of the first to be addressed and is still one of the core problems. As the name suggest, image denoising deals with the removal of noise, which is unavoidably associated to the acquisition process, from a digital image. Several types of noise follow from the image acquisition and their sources vary: shot noise due to the photon count variations, which is i itself a random process (with Poisson distribution [11]), thermal noise from the electronic components, and sensor noise due to the incoming light, where too much light can lead to saturation and too few causes the sensors to produce unreliable measurements. Techniques to lessen this undesirable consequence require a significant computational power and, as such, only in the last years the progress has been more momentous. This dissertation proposes a new method for image denoising.

1.2 Objectives

The objectives for this course are the following:

- Provide a framework for the dissertation's theme;
- Review the state of the art on image denoising;
- Define the methodologies to be used;
- Plan and schedule the work to be done during the dissertation course;

1.3 Framework

Signal processing is an important and current topic studied in scope of Electrical and Computer Engineering and the range of the applications in this area is immense. A particular subject of signal processing is image processing. As the name suggest, the input signal is an image, be it a photograph (2D signal) or a video (3D signal), and the output is also an image which has been modified in some way. This modification is in the sense of improving pictorial information or processing data for autonomous machine interpretation. Image denoising (removal of noise), image deblurring (removal of a blur, for example due to out-of-focus camera or motion), image inpainting (reconstruction of lost or deteriorated parts, due to coding errors or saturation), image segmentation (partitioning of the image into its constituent parts or objects), super-resolution (improving image resolution) are some of the examples of the problems that image processing is concerned with; in this thesis, the denoising problem will be addressed. All of the above can be seen as inverse problems meaning that there is some observed data and the objective is to determine a set of parameters which originated it, according to a theoretical model, different for each type of problem. Typically, these problems are ill-posed in the sense that there is not an unique solution.

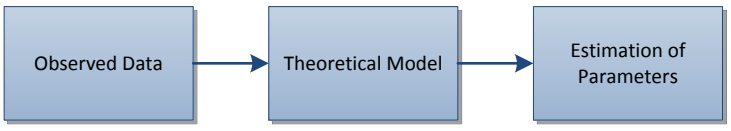


Figure 1.1: Inverse Problem.

It was suggested in [8] that state of the art image denoising algorithms are almost at the theoretical limit of what can be achieved in terms of peak signal-to-noise ratio. This dissertation proposes another method for image denoising based on a Gaussian mixture model, obtained from the noisy image itself, in the hope of producing results comparable with the recent state of the art not only in terms of PSNR but also in visual quality.

1.4 Problem Statement

As mentioned above, image denoising is one of the core topics of image processing. More specifically, it is an inverse problem where the objective is to determine the true value of a signal, knowing that it has been corrupted by noise. Mathematically, the theoretical model for this problem can be put in very simple terms as

$$\mathbf{y} = \mathbf{x} + \mathbf{w} \tag{1.1}$$

where **y** denotes the input signal, **x** the clean image that we want to estimate and **w** is noise. A particularization of the diagram of figure 1.1 to the denoising problem is illustrated in figure 1.2.

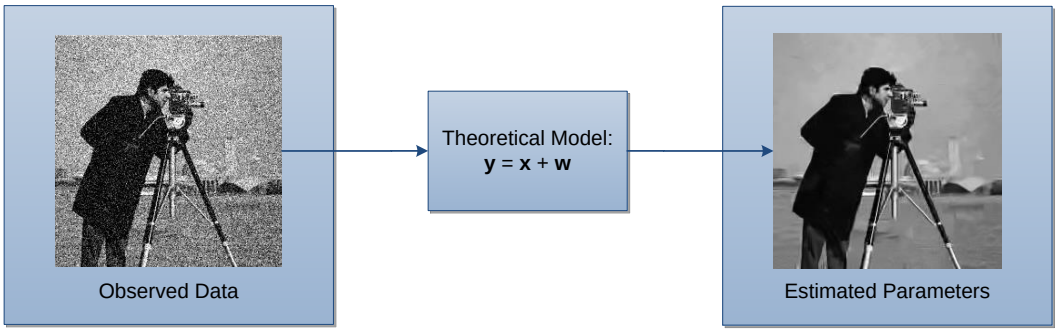


Figure 1.2: Denoising Problem.

It is important to note that without making any assumption about the true signal, this problem is impossible to solve satisfactorily. A simplifying hypothesis that is usually made is to assume that **w** is white noise with known distribution $\mathcal{N} \sim (0, \sigma^2\mathbf{I})$. This assumption is not restrictive to the problem at hand since it only shortcuts an additional step of estimating the noise from the observed image itself, [12] [22] [17].

One of the most common approaches to solve this problem, due to its tractability, is to estimate the image that maximizes the *a posteriori* probability (MAP estimator)

$$\hat{\mathbf{x}} = \mathbf{x}_{MAP} = \underset{\mathbf{x}}{arg\,max} \quad p_{\mathbf{X}|\mathbf{Y},\theta}(\mathbf{x} \mid \mathbf{y}, \theta) \tag{1.2}$$

$$= \underset{\mathbf{x}}{arg\,max} \quad \frac{p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y} \mid \mathbf{x})\, p_{\mathbf{X}}(\mathbf{x})}{p_{\mathbf{Y}}(\mathbf{y})} \tag{1.3}$$

$$= \underset{\mathbf{x}}{arg\,max} \quad p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y} \mid \mathbf{x})\, p_{\mathbf{X}}(\mathbf{x}) \tag{1.4}$$

yet this is not the most suitable. In this notation, $p_{\mathbf{Y}|\mathbf{X},\theta}(\mathbf{y} \mid \mathbf{x}, \theta)$ corresponds to the probability density function of the random variable **Y**, conditioned to the true image and noise parameters, commonly called the *observation model* in signal processing. Furthermore, equation (1.3) follows from equation (1.2) by application of the Bayes rule which relates the posterior probability with the *a priori* probability, or likelihood, and a *prior*. Note that the noise parameters, θ, where dropped due to simplifying assumption mentioned above and overloading the notation so that **Y** encompasses both the observed image and

noise parameters.

The minimum mean squared error corresponds to another quality measure for the estimator

$$\begin{aligned}\hat{\mathbf{x}} = \mathbf{x}_{MMSE} &= \mathbb{E}[\mathbf{x} \mid \mathbf{y}] \\ &= \int \mathbf{x}\, p_{\mathbf{x}|\mathbf{y}}(\mathbf{x} \mid \mathbf{y}) d\mathbf{x} \\ &= \int \mathbf{x}\, \frac{p_{\mathbf{y}|\mathbf{x}}(\mathbf{y} \mid \mathbf{x})\, p_{\mathbf{x}}(\mathbf{x})}{p_{\mathbf{y}}(\mathbf{y})} d\mathbf{x}\end{aligned} \tag{1.5}$$

but, in general, it is expensive to calculate the factor in the denominator. Besides being intuitively more adequate than the MAP, this estimator is also advantageous since, after computing the estimator, its quality is usually measured by the PSNR (peak signal-to-noise ratio), given by

$$PSNR = 10 \log_{10} \left(\frac{1}{MSE} \right), \tag{1.6}$$

where the MSE denotes the mean square error with respect to the true values of the noise free image. Note that this measure is only applicable to synthetic images since otherwise the noise free image is not known. The results are also compared with respect to their visual quality but this measure lacks objectivity and is impossible to determine resorting to a computer, although there have been many proposals of quantitative measures of visual quality [reference].

Chapter 2

State-of-the-art

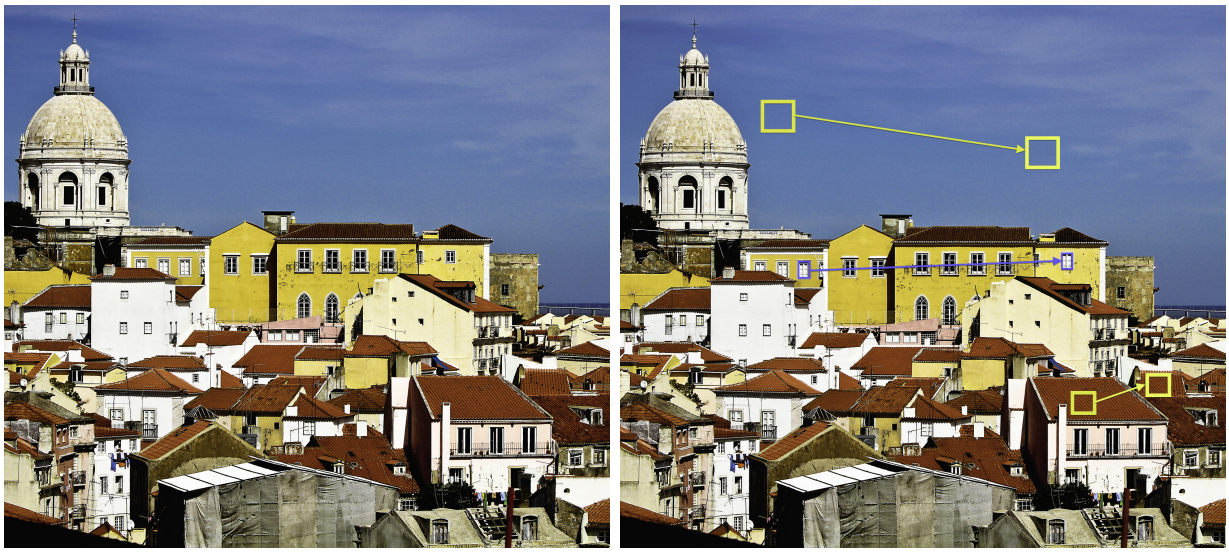
There are essentially three paradigms in modern image denoising: global methods, filtering methods, and patch-based methods. The first refers, for example, to methods that apply some kind of transformation to the noisy image, as a whole, and then work with the coefficients. Methods based on wavelet (or other multi-scale transforms) thresholding [6], total-variation [5] or PCA [7] are examples of global methods. The second paradigm [18], used for example in Adobe Photoshop, repeatedly applies a filter or a sequence of filters to the noisy image and then combines the results to obtain a cleaner image. The last paradigm (patch-based methods) will be investigated with greater detail since it encompasses the proposed algorithm. Henceforth, only methods from this class will be referred.

This section presents and compares several patch-based methods corresponding to the state of the art in denoising images corrupted by additive white Gaussian noise, both in terms of peak signal-to-noise ratio and visual quality. Furthermore, we can divide these methods into three categories: non-local means, sparse representations using a dictionary, probabilistic models, namely those based on mixtures. Examples of algorithms belonging to each of these families are provided below.

Patch-based methods explore the redundancy of the image, both local and non-local and are based on the fact that in natural images the pixels are positively correlated between each other: for example, an edge is always an abrupt change in brightness, regardless of the orientation and the intensity difference in brightness itself. This characteristic is called *self-similarity*. Another way to interpret this feature is to consider that it is possible to alter the image by changing the location of small patches without changing the overall image as illustrated in figures 2.1a and 2.1b.

This knowledge allows us to determine a model of each pixel from its neighbourhood, *i.e.* in the whole image there are enough pixels, *j*, whose neighbourhood is similar to that of the current pixel *i*, according to some metric, such that it is possible to recreate a new denoised image. The similar pixels can be obtained either from the noisy image itself or from an external database (set of clean images). The proposed method focuses on the former.

Before describing the algorithms it is important to define what a patch is, as well as some of the parameters that have impact on the results. A patch corresponds to a small neighbourhood around a given pixel, of size $k \times k$. The patch size is thus an important parameter because the bigger the patch



(a) Original Image. (b) Edited Image.

Figure 2.1: Image *self-similarity*. Source: `http://www.visitlisboa.com/`

the less likely it is to find another patch similar to the first. Yet, due to the presence of additive noise, the minimum patch size (3×3) may not be adequate. In fact, experimental results show that for the typical values of noise standard deviation ($\sigma \in [20, 40]$) the best results are obtain for 8×8 patches and this size may not be sufficient for higher σ. Another important parameter is the step when going from one patch to another. If the step is unitary (in either direction) all the patches are analysed which obviously is computationally expensive but ensures that the ones that are more resembling are found. Also, if the step is smaller than the patch size it may be necessary to add an additional stage to the algorithm in order to combine the various estimations of each pixel due to patch overlap. Figure 2.2 illustrates both of this parameters.

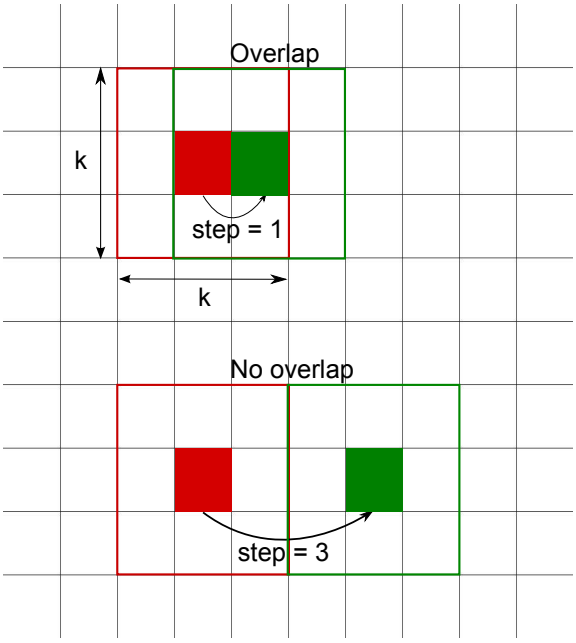


Figure 2.2: Patch-based methods' parameters: patch size and step.

2.1 NL-Means

This method propelled the patch-based paradigm for image denoising [3]. In its simplest form, this method searches the noisy image for patches similar to the current one and applies a weighted average, where the weights are proportional to the similarity metric. As mentioned before, this method takes advantage of the high degree of redundancy in natural images. Intuitively, this assumption is reasonable if we consider the immediate neighbourhood of the each patch yet this can still be found non-locally.

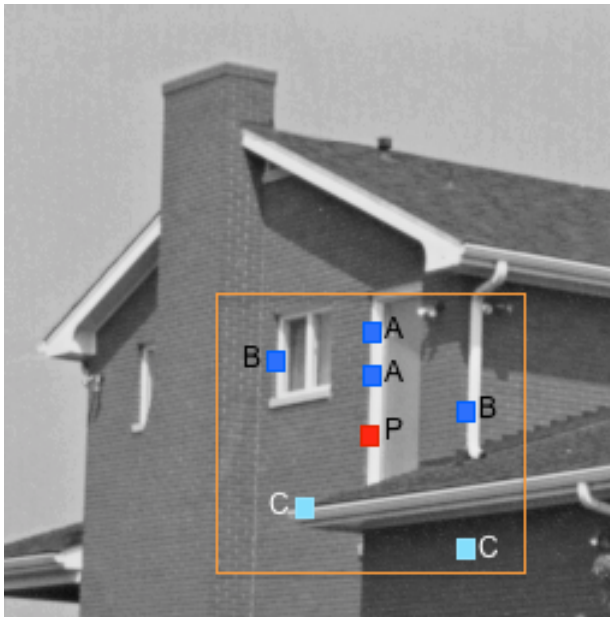


Figure 2.3: Non-local means algorithm.

Figure 2.3 illustrates the behaviour of the algorithm where patches *A* and *B* resemble much more the current patch, *P*, than patches *C*. Since exploring the whole image can be costly, a search window is defined. Naturally, this restriction influences the outcome of the procedure since perchance the most similar patches can be found outside this area thus defining an important trade-off.

[...]

2.2 BM3D

Arguably the best method for image restoration and one of the best for image denoising, BM3D [9] [15] exploits the fact that images may have a sparse representation in the transform domain, meaning that there are only a few significant coefficients and that the signal can be well approximated by a linear combination of a small number of basis elements.

There are three major steps in this algorithm: Grouping, Collaborative Filtering and Aggregation. In the first step, for a given patch, the noisy image is searched for other similar fragments. Then, these patches are stacked, forming a 3D block called Group. Follows the second step, where the denoising *per se* is made. A linear transformation is applied to the 3D block and then the coefficients are shrunk, such that the lower coefficients, containing less of the signal's energy (noise), are set to zero. Afterwards, the

inverse transform is applied in order to obtain the denoised image. Since the denoised patches overlap there's still a need to combine the various estimations of each pixel. This is the last step of the algorithm (Aggregation) which applies a weighted average to the estimated denoised patches.

All this process is repeated a second time with two differences: each patch is now compared with the filtered patches instead of the noisy ones and, rather than using a simple threshold criteria in the coefficient shrinkage step, it employs a Wiener filter. This second step improves significantly the obtained results although further repetition only brings incremental changes.

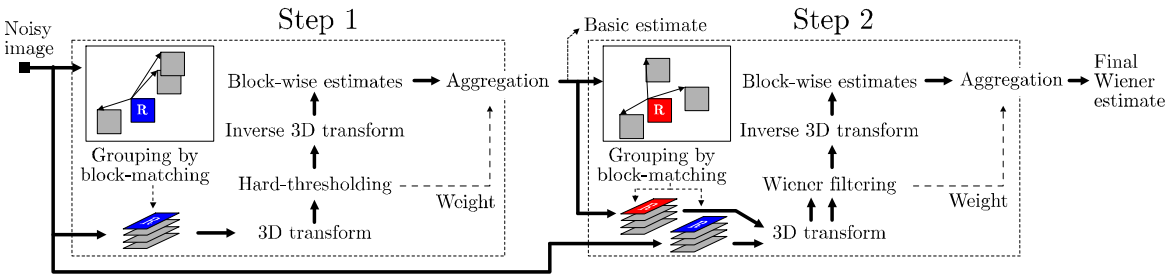


Figure 2.4: Graphical representation of the BM3D Algorithm. Source: [9].

2.3 K-SVD

This algorithm [1][21][16] is based on sparse coding. It tries to identify a small set of patterns that can be combined to reproduce each patch of the original (denoised) input: $\mathbf{x} \approx D\alpha$. These patterns are sometimes referred to as atoms establishing an analogy with chemical elements: combining patterns (atoms) from a dictionary (periodic table) to obtain an image (molecule). K-SVD can be divided in two steps: first, using the current dictionary, the image is decomposed into a set of coefficients ("atom decomposition") and, second, the dictionary or codebook is updated, assuming constant and known coefficients, in order to better fit the input signal. In the end, an aggregation technique is applied as well to combine the sparse representations of each patch. Mathematically, the algorithm can be formulated as

$$\underset{\mathbf{x},\mathbf{D}}{\hat{\mathbf{x}} = arg\,min} \; || \mathbf{y} - \mathbf{D}\mathbf{x} ||_F^2 \qquad \text{subject to} \qquad || \mathbf{x}_i ||_0 \leq T_0 \tag{2.1}$$

where the constraint enforces sparsity of the coefficients since the l_0-norm counts the number of non-zero entries.

The first step is done using a pursuit algorithm and regards the minimization of equation (2.1) with respect to $\mathbf{x}$, *i.e.* finding the best combinations of patterns from the dictionary to represent each patch. This can be interpreted as a clustering algorithm which generalizes the k-means. Next, the dictionary is updated, one column at a time. Since this is done sequentially, the result is obtained resorting to Singular Value Decomposition. Hence the name of the algorithm.

Moreover, in the beginning of the algorithm, the dictionary must be initialized. This can be done using

a set of noise free patches obtained from a database, patches from the noisy image or an orthogonal basis such as wavelet, curvelet, discrete cosine transform among other possibilities.

2.4 Mixture Model

The last class of patch-based methods tries to find a probabilistic model for the noise-free patches using mixtures. The proposed method belongs to this family of methods and as such ought to be examined more thoroughly.

A mixture model is a flexible tool to model univariate or multivariate data with numerous applications in areas such as image analysis, pattern recognition, and machine learning. Given a set of observed samples (training) it is possible to infer properties of the density function that originated those samples. A mixture model for a random vector, **x**, has the form

$$p(\mathbf{x} \mid \phi) = \sum_{i=1}^{k} \alpha_i f(\mathbf{x} \mid \phi_i), \tag{2.2}$$

where $f(\mathbf{x} \mid \phi_i)$ are the density functions of the mixture components (same density function for all the components but with different parameters, ϕ), α_i represent the weights of each component, summing to one.

Although a mixture model could, in principle, use completely arbitrary distributions, most commonly the mixture model uses a combination of Gaussians mainly due to its capability to approximate any continuous density function as well as analytical tractability. In this situation, equation (2.2) becomes equation (2.3).

$$p(\mathbf{x} \mid \phi) = \sum_{i} \alpha_i \mathcal{N}(\mathbf{x}; \mu_i, C_i) \tag{2.3}$$

Let **X** denote a d-dimensional random variable and **x** a particular realization. Given a set of independent, identically distributed samples $\mathcal{X} = \{\mathbf{x}^0, \mathbf{x}^1, \ldots, \mathbf{x}^n\}$, the likelihood function of a k-component Gaussian mixture is written in equation (2.4).

$$\mathcal{L}(\phi, \alpha, \mathcal{X}) = p(\mathcal{X} \mid \phi, \alpha) = \prod_{i=1}^{n} p(\mathbf{x}^{(i)} \mid \phi) = \sum_{i=1}^{n} \log \sum_{j=1}^{k} \alpha_j \mathcal{N}(\mathbf{x}^{(i)}; \phi_j) \tag{2.4}$$

$$(\hat{\phi}, \hat{\alpha})_{ML} = \underset{\phi,\alpha}{arg\,max} \quad \mathcal{L}(\phi, \alpha, \mathcal{X}) \tag{2.5}$$

$$(\hat{\phi}, \hat{\alpha})_{MAP} = \underset{\phi,\alpha}{arg\,max} \quad p(\phi, \alpha) + \mathcal{L}(\phi, \alpha, \mathcal{X}) \tag{2.6}$$

The estimation of the parameters of each mixture component using the maximum likelihood equation (2.5)) or maximum a posteriori method (equation (2.6)) can not be done analytically. Yet, efficient algorithms have been developed in order to fit the mixture model to the observed data, in an iterative way. An example of such algorithm is the widely acknowledged *Expectation-Maximization*, abbreviated

EM, algorithm. The derivation of this algorithm can be found in the Appendix (B). Consider also, a set of hidden variables $\mathcal{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n\}$, binary, with $\mathbf{z}_i \in \mathrm{R}^k$, where entry s is 1 if $\mathbf{x}_i$ was generated with component s ($\mathcal{N}(\mathbf{x}_i; \mu_s, C_s)$) and 0 otherwise. Equation (2.4) becomes:

$$\mathcal{L}_{complete}(\phi, \alpha, \mathcal{X}, \mathcal{Z}) = p(\mathcal{X}, \mathcal{Z} \mid \phi, \alpha) = \prod_{i=1}^{n} p(\mathbf{x}^{(i)}, \mathbf{z}^{(i)} \mid \phi, \alpha) = \sum_{i=1}^{n} \sum_{j=1}^{k} z_i^{(j)} \log(\alpha_j \mathcal{N}_j(\mathbf{x}^{(i)}; \phi_j)) \tag{2.7}$$

Applying the E-step to the Gaussian mixture model:

$$Q((\phi, \alpha), (\hat{\phi}^t, \hat{\alpha}^t)) = \mathbb{E}[\mathcal{L}_{complete}(\phi, \alpha, \mathcal{X}, \mathcal{Z}) \mid \mathcal{X}, \hat{\phi}^t, \hat{\alpha}^t] = \mathcal{L}_{complete}(\phi, \alpha, \mathcal{X}, \mathbb{E}[\mathcal{Z} \mid \mathcal{X}, \hat{\phi}^t, \hat{\alpha}^t]) \tag{2.8}$$

where

$$\begin{aligned} \omega_i^s \equiv \mathbb{E}[\mathbf{z}_i^s \mid \mathcal{X}, \hat{\phi}^t, \hat{\alpha}^t]) &= Pr[z_i^s = 1 \mid \mathcal{X}, \hat{\phi}^t, \hat{\alpha}^t] \\ &= \frac{\hat{\alpha}_s^t \mathcal{N}(\mathbf{x}_i; \hat{\phi_s}^t)}{\sum_{r=1}^{k} \hat{\alpha}_r^t \mathcal{N}(\mathbf{x}_i; \hat{\phi_r}^t)} \end{aligned} \tag{2.9}$$

and then the M-step to obtain:

$$\hat{\alpha_s}^{t+1} = \frac{1}{N} \sum_{i=1}^{N} \omega_i^{(s)} \tag{2.10}$$

$$\hat{\mu_s}^{t+1} = \frac{\sum_{i=1}^{N} \mathbf{x}_i \omega_i^{(s)}}{\sum_{i=1}^{N} \omega_i^{(s)}} \tag{2.11}$$

$$\hat{C_s}^{t+1} = \frac{\sum_{i=1}^{N} (\mathbf{x}_i - \hat{\mu_s}^{t+1})(\mathbf{x}_i - \hat{\mu_s}^{t+1})^T \omega_i^{(s)}}{\sum_{i=1}^{N} \omega_i^{(s)}} \tag{2.12}$$

An advantage that follows from this method is that computing the minimum mean squared error estimator, which is the desired one, is quite simple. Taking into account the theoretical model, (1.1), and the assumption that the noise is Gaussian with distribution $\mathcal{N} \sim (0, \sigma^2 \mathbf{I})$ we get

$$p_{\mathbf{x}}(\mathbf{x}) = \sum_i \alpha_i \mathcal{N}(\mathbf{x}; \mu_i, C_i) \tag{2.13}$$

$$p_{\mathbf{y}}(\mathbf{y}) = \sum_i \alpha_i \mathcal{N}(\mathbf{y}; \mu_i, C_i + \sigma^2 \mathbf{I}) \tag{2.14}$$

$$p_{\mathbf{y}|\mathbf{x}}(\mathbf{y} \mid \mathbf{x}) = \mathcal{N}(\mathbf{y}; \mathbf{x}, \sigma^2 \mathbf{I}) \tag{2.15}$$

$$\begin{aligned} p_{\mathbf{x}|\mathbf{y}}(\mathbf{x} \mid \mathbf{y}) &= \frac{p_{\mathbf{y}|\mathbf{x}}(\mathbf{y} \mid \mathbf{x})\, p_{\mathbf{x}}(\mathbf{x})}{p_{\mathbf{y}}(\mathbf{y})} \\ &= \frac{p_{\mathbf{y}|\mathbf{x}}(\mathbf{y} \mid \mathbf{x})\, p_{\mathbf{x}}(\mathbf{x})}{\int p_{\mathbf{y}|\mathbf{x}}(\mathbf{y} \mid \mathbf{x})\, p_{\mathbf{x}}(\mathbf{x}) d\mathbf{x}} \end{aligned}$$

and knowing that the product of two Gaussian distributions is proportional to a Gaussian distribution

[19], for each component i we obtain

$$\begin{aligned} p_{\mathbf{x}}(\mathbf{x})\, p_{\mathbf{y}|\mathbf{x}}(\mathbf{y} \mid \mathbf{x}) &= \alpha_i \mathcal{N}(\mathbf{x}; \mu_i, C_i) \times \mathcal{N}(\mathbf{y}; \mathbf{x}, \sigma^2 \mathbf{I}) \\ &= \alpha_i \frac{1}{\sqrt{det((2\pi)^d C_i)(2\pi)^d \sigma^2}} \times \exp\left(-\frac{1}{2}(\mathbf{x}-\mu_i)^T C_i^{-1}(\mathbf{x}-\mu_i)\right) \times \exp\left(-\frac{1}{2}\frac{(\mathbf{x}-\mathbf{y})^T(\mathbf{x}-\mathbf{y})}{\sigma^2}\right) \\ &= \alpha_i^{'} \times \frac{1}{\sqrt{det((2\pi)^d C_i^{'})}} \times \exp\left(-(\mathbf{x}-\mu_i^{'})^T (C_i^{'})^{-1}(\mathbf{x}-\mu_i^{'})\right) \end{aligned}$$

where

$$\alpha_i^{'}(\mathbf{y}) = \alpha_i \frac{\sqrt{det((2\pi)^d \left(C_i^{-1} + (\sigma^2\mathbf{I})^{-1}\right)^{-1})}}{\sqrt{det((2\pi)^d C_i)((2\pi)^d \sigma^2) det((2\pi)^d (C_i + \sigma^2 \mathbf{I}))}} \times \exp\left(-\frac{1}{2}(\mathbf{y}-\mu_i)^T (C_i + \sigma^2\mathbf{I})^{-1}(\mathbf{y}-\mu_i)\right) \tag{2.16}$$

$$\mu_i^{'}(\mathbf{y}) = \left(C_i^{-1} + (\sigma^2\mathbf{I})^{-1}\right)^{-1} \left(C_i^{-1}\mu_i + (\sigma^2\mathbf{I})^{-1}\mathbf{y}\right) \tag{2.17}$$

$$C_i^{'}(\mathbf{y}) = \left(C_i^{-1} + (\sigma^2\mathbf{I})^{-1}\right)^{-1} \tag{2.18}$$

Applying these results to the generic expression for the estimator, (1.5), and noting that the denominator does not depend on **x**

$$\begin{aligned} \mathbf{x}_{MMSE} &= \int \mathbf{x} \frac{p_{\mathbf{y}|\mathbf{x}}(\mathbf{y} \mid \mathbf{x})\, p_{\mathbf{x}}(\mathbf{x})}{p_{\mathbf{y}}(\mathbf{y})} d\mathbf{x} \\ &= \int \mathbf{x} \frac{p_{\mathbf{y}|\mathbf{x}}(\mathbf{y} \mid \mathbf{x})\, p_{\mathbf{x}}(\mathbf{x})}{\int p_{\mathbf{y}|\mathbf{x}}(\mathbf{y} \mid \mathbf{x})\, p_{\mathbf{x}}(\mathbf{x}) d\mathbf{x}} d\mathbf{x} \\ &= \int \mathbf{x} \sum_{i=1}^{k} \alpha_i^{'}(\mathbf{y}) \mathcal{N}(\mathbf{x}; \mu_i^{'}(\mathbf{y}), C_i^{'}(\mathbf{y})) d\mathbf{x} \\ &= \sum_{i=1}^{k} \alpha_i^{'}(\mathbf{y}) \int \mathbf{x} \mathcal{N}(\mathbf{x}; \mu_i^{'}(\mathbf{y}), C_i^{'}(\mathbf{y})) d\mathbf{x} \\ &= \sum_{i=1}^{k} \alpha_i^{'}(\mathbf{y}) \mu_i^{'}(\mathbf{y}) \end{aligned} \tag{2.19}$$

2.5 Algorithm comparison

[INSERT FIGURES]

sigma (σ)	Lena			Cameraman			House		
	BM3D	K-SVD	NL-Means	BM3D	K-SVD	NL-Means	BM3D	K-SVD	NL-Means
5	38.72	38.53	37.97	38.29	37.97	37.51	39.83	39.47	38.67
10	35.93	35.55	34.39	34.18	33.76	33.49	36.61	36.05	35.01
15	34.27	33.74	32.08	31.91	31.54	31.15	34.94	34.41	32.71
20	33.05	32.40	31.59	30.48	30.07	29.87	33.77	33.21	32.40
25	32.08	31.34	30.51	29.45	28.94	28.71	32.86	32.21	31.10
30	31.26	30.46	29.52	28.64	28.12	27.96	32.09	31.25	30.01
35	30.56	29.69	28.98	27.93	27.42	27.07	31.38	30.44	29.61
40	29.86	29.02	28.26	27.18	26.79	26.58	30.65	29.56	28.82
50	29.05	27.83	27.22	26.12	25.76	25.29	29.69	28.02	27.40
70	27.57	26.13	25.48	24.61	23.86	23.42	27.91	25.44	25.19
100	25.95	24.54	23.80	23.07	21.56	21.37	25.87	23.64	23.08

Table 2.1: Algorithm comparison in terms of PSNR (dB) for three different grayscale images.

Chapter 3

Future Work

In this chapter, it is presented the practical work that will be developed in the dissertation course to be carried out on the second semester of 2013/2014. By all means, the planning and scheduling conferred below is not definite and will, most likely, be altered along the way. Nevertheless, it provides a valuable help in the organization of the semester.

3.1 Improvements on the Gaussian Mixture Model algorithm

In this section are presented some of the improvements that will be made to the algorithm presented in [4] and that will lead to more satisfying results and a more robust method.

Firstly, whereas in synthetic images we know the noise variance, in real images the noise is unknown. Consequently, an algorithm to estimate the noise [include reference] must be included in the method.

Secondly, the number of mixture components will be estimated automatically, in an adaptive way, using the algorithm described in [10].

Thirdly, the aggregation technique to deal with the overlapping estimates will be a weighted average, considering the variance of each mixture component.

Lastly, a few tricks that, in general, improve the performance of every algorithm, will be implemented [14]. Such techniques include oracle filtering, multi-scale search and a particular way of dealing with flat areas. The former consists on reiterating all the denoising process but, instead of comparing each patch with the noisy patches, comparing with the clean image estimate from the first iteration which, in principle, is closer to the true noise free image. Note that in this second iteration it is still the noisy image that is being denoised although the comparison is made with respect to the results of the first iteration. Multi-scale search refers to the use of the original noisy image as well as some low resolution versions when looking for similar patches. This technique is based on the assumption that the self-similarity characteristic of natural images is preserved at different scales. Furthermore, another kind of multi-scale search that will be implemented regards the use of different patch sizes when looking for identical patches and then choosing the ones that resemble the most. The last technique is used to avoid artifacts that appear in the denoised image flat areas when the noise values are high. By definition, flat areas are

those where the variance is close to zero thus, when noise is added, the new variance of the flat area is approximately that of the noise.

3.2 Planning and Scheduling

3.3 Conclusions

Appendix A

Estimation Theory

This chapter regards some of the concepts necessary to solve the problem at hand. When dealing with estimation problems there are, essentially, three widely acknowledged loss functions [13]: ”0/1”, quadratic error and absolute error. In the following, only the first two loss functions will be investigated since they lead to the maximum *a posteriori* and minimum mean squared error estimators, respectively. Moreover, a Bayesian decision rule will be considered, as opposed to a frequentist one and with this approach the criterion for the evaluation of the decision rule will be the *a posteriori* expected loss. The following derivations are applied to equation (1.1).

A.1 Bayesian Inference

Let $L(\mathbf{x}, a)$ denote the loss function which measures the cost of an action, a, with respect to the true value $\mathbf{x}$. The Bayesian decision rule can be formulated as

$$\begin{aligned} \hat{\mathbf{x}} &= \underset{a}{argmin} \quad \mathbb{E}_{\mathbf{X}}[L(\mathbf{x}, a) \mid \mathbf{y}] \\ &= \underset{a}{argmin} \int L(\mathbf{x}, a)\, p_{\mathbf{x}|\mathbf{y}}(\mathbf{x} \mid \mathbf{y}) d\mathbf{x}, \end{aligned} \tag{A.1}$$

where the notation $\mathbb{E}_{\mathbf{X}}[\bullet \mid \mathbf{y}]$ stands for the expected value, with respect to the random variable $\mathbf{X}$, given the observed samples, $\mathbf{y}$.

A.1.1 ”0/1”

The ”0/1” ϵ-loss function is defined, for the vectorial case, as

$$L_\epsilon(\mathbf{x}, \mathbf{a}) = \begin{cases} 1 & \mid \mathbf{x} - \mathbf{a} \mid \geq \epsilon \\ 0 & \mid \mathbf{x} - \mathbf{a} \mid < \epsilon \end{cases} \tag{A.2}$$

which, substituting in the decision rule (A.1) gives

$$
\begin{aligned}
\hat{\mathbf{x}} &= \underset{\mathbf{a}}{arg\,min} \int L_\epsilon(\mathbf{x}, \mathbf{a})\, p_{\mathbf{x}|\mathbf{y}}(\mathbf{x} \mid \mathbf{y}) d\mathbf{x} \\
&= \underset{\mathbf{a}}{arg\,min} \int_{-\infty}^{a-\epsilon} p_{\mathbf{x}|\mathbf{y}}(\mathbf{x} \mid \mathbf{y}) d\mathbf{x} + \int_{a+\epsilon}^{\infty} p_{\mathbf{x}|\mathbf{y}}(\mathbf{x} \mid \mathbf{y}) d\mathbf{x} \\
&= \underset{\mathbf{a}}{arg\,min} \quad 1 - \int_{a-\epsilon}^{a+\epsilon} p_{\mathbf{x}|\mathbf{y}}(\mathbf{x} \mid \mathbf{y}) d\mathbf{x} \\
&= \underset{\mathbf{a}}{arg\,max} \quad \int_{a-\epsilon}^{a+\epsilon} p_{\mathbf{x}|\mathbf{y}}(\mathbf{x} \mid \mathbf{y}) d\mathbf{x}
\end{aligned} \tag{A.3}
$$

and taking the limit when ϵ goes to zero becomes the maximum *a posteriori* estimator

$$
\hat{\mathbf{x}} = \mathbf{x}_{MAP} = \underset{x}{arg\,max} \quad p_{\mathbf{x}|\mathbf{y}}(\mathbf{x} \mid \mathbf{y}) \tag{A.4}
$$

This estimator is not the most adequate since it attributes a constant loss independently of the discrepancy (bigger than ϵ) between the estimation and the true value. Yet, this estimator is, in general, applied due to its tractability.

A.1.2 Quadratic Error

The quadratic error loss function is defined, for the vectorial case, as

$$
L(\mathbf{x}, \mathbf{a}) = ||\mathbf{x} - \mathbf{a}||^2 = (\mathbf{x} - \mathbf{a})^T (\mathbf{x} - \mathbf{a}) \tag{A.5}
$$

which, substituting in the decision rule (A.1) gives

$$
\begin{aligned}
\hat{\mathbf{x}} &= \underset{\mathbf{a}}{arg\,min} \int L(\mathbf{x}, \mathbf{a})\, p_{\mathbf{x}|\mathbf{y}}(\mathbf{x} \mid \mathbf{y}) d\mathbf{x} \\
&= \underset{\mathbf{a}}{arg\,min} \int (\mathbf{x} - \mathbf{a})^T (\mathbf{x} - \mathbf{a})\, p_{\mathbf{x}|\mathbf{y}}(\mathbf{x} \mid \mathbf{y}) d\mathbf{x} \\
&= \underset{\mathbf{a}}{arg\,min} \int (\mathbf{x}^T\mathbf{x} - 2\mathbf{x}^T\mathbf{a} + \mathbf{a}^T\mathbf{a})\, p_{\mathbf{x}|\mathbf{y}}(\mathbf{x} \mid \mathbf{y}) d\mathbf{x} \\
&= \underset{\mathbf{a}}{arg\,min} \int \mathbf{x}^T\mathbf{x}\, p_{\mathbf{x}|\mathbf{y}}(\mathbf{x} \mid \mathbf{y}) d\mathbf{x} - 2\mathbf{a}^T \int \mathbf{x}\, p_{\mathbf{x}|\mathbf{y}}(\mathbf{x} \mid \mathbf{y}) d\mathbf{x} + \mathbf{a}^T\mathbf{a} \int p_{\mathbf{x}|\mathbf{y}}(\mathbf{x} \mid \mathbf{y}) d\mathbf{x} \\
&= \underset{\mathbf{a}}{arg\,min} \quad \mathbb{E}[\mathbf{x}^T\mathbf{x} \mid \mathbf{y}] - 2\mathbf{a}^T \mathbb{E}[\mathbf{x} \mid \mathbf{y}] + \mathbf{a}^T\mathbf{a}
\end{aligned} \tag{A.6}
$$

Taking the derivative of (A.6) with respect to the *action*, $\mathbf{a}$, and setting to zero leads to

$$
\hat{\mathbf{x}} = x_{MMSE} = \mathbb{E}[\mathbf{x} \mid \mathbf{y}] \tag{A.7}
$$

Intuitively, this estimator is more appropriate because it computes the distance between the estimate and the true value. However, this estimator is much harder to calculate, this being the main reason why it is only used in some specific situations.

Appendix B

Expectation-Maximization

The Expectation-Maximization algorithm is a well-studied method [references] and very useful to determine the maximum likelihood or maximum *a posteriori* estimators of the parameters of an underlying distribution, when there is unobserved data or the computation of the estimators becomes simplified if we assume the existence of a set of hidden variables.

B.1 Algorithm

Consider a set of observed random variables **y** and another set **x** which is unobserved (hidden). Furthermore, consider a set of parameters θ that we want to estimate. The *a posteriori* probability function $p_{\theta|\mathbf{y}}(\theta \mid \mathbf{y})$ is then proportional to $p_{\mathbf{y}|\theta}(\mathbf{y} \mid \theta)\, p_\theta(\theta)$ or to $p_{\mathbf{x},\mathbf{y}|\theta}(\mathbf{x}, \mathbf{y} \mid \theta)\, p_\theta(\theta)$ where $p_{\mathbf{y}|\theta}(\mathbf{y} \mid \theta)$ is related to $p_{\mathbf{x},\mathbf{y}|\theta}(\mathbf{x}, \mathbf{y} \mid \theta)$ through marginalization:

$$p_{\mathbf{y}|\theta}(\mathbf{y} \mid \theta) = \int p_{\mathbf{x},\mathbf{y}|\theta}(\mathbf{x}, \mathbf{y} \mid \theta) d\mathbf{x} \tag{B.1}$$

The usefulness of the EM algorithm is due to the fact that it allows $p_{\theta|\mathbf{y}}(\theta \mid \mathbf{y})$ to be computed without direct manipulation of the marginal distribution $p_{\mathbf{y}|\theta}(\mathbf{y} \mid \theta)$.

To do so, the algorithm is then divided into two steps, applied alternately: *Expectation* (E) and *Maximization* (M). Being an iterative procedure, it requires an initialization, θ^0, which considerably influences the outcome (see [references] for ways to mitigate this dependency). The iterations stop when some criteria is met.

B.1.1 E-Step

The first step is the *Expectation*-step and, as the name suggest, it computes an expected value. Let $Q(\theta, \theta^i)$ denote an auxiliary value corresponding to the conditional expectation, with respect to the unobserved random variables, **x**, of the logarithm of the complete *a posteriori* probability function[1] given the observed random variables and the current estimator θ^i

[1] The use of the logarithm of the function does not influence the estimation in any way because it is monotonically increasing and achieves its maximum value at the same place as the function itself.

$$
\begin{aligned}
Q(\theta, \theta^i) &= \mathbb{E}[log\left(p_{\mathbf{x},\mathbf{y}|\theta}(\mathbf{x}, \theta \mid \mathbf{y})\right) \mid \mathbf{y}, \theta^i] \\
&\propto \mathbb{E}[log\left(p_{\mathbf{x},\mathbf{y}|\theta}(\mathbf{x}, \mathbf{y} \mid \theta)\, p_\theta(\theta)\right) \mid \mathbf{y}, \theta^i] \\
&= log\left(p_\theta(\theta)\right) + \int p_{\mathbf{x}|\mathbf{y},\theta}(\mathbf{x} \mid \mathbf{y}, \theta^i)\, log\left(p_{\mathbf{x},\mathbf{y}|\theta}(\mathbf{x}, \mathbf{y} \mid \theta)\right) d\mathbf{y}
\end{aligned}
\tag{B.2}
$$

[...]

B.1.2 M-Step

The *Maximization*-step corresponds to updating the current values of the parameters, θ^i, in such a way that the conditional expectation of the *E*-step is maximum

$$
\theta^{i+1} = \underset{\theta}{arg\,max} \quad Q(\theta, \theta^i) \tag{B.3}
$$

$$
= \underset{\theta}{arg\,max} \quad log\left(p_\theta(\theta)\right) + \int p_{\mathbf{x}|\mathbf{y},\theta}(\mathbf{x} \mid \mathbf{y}, \theta^i)\, log\left(p_{\mathbf{x},\mathbf{y}|\theta}(\mathbf{x}, \mathbf{y} \mid \theta)\right) d\mathbf{y} \tag{B.4}
$$

If the prior $p_\theta(\theta)$ is flat equation (B.4) becomes the maximum likelihood estimator otherwise it is the maximum *a posteriori*.

[...]

Bibliography

[1] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation. *Signal Processing, IEEE Transactions on*, 54(11):4311–4322, November 2006. ISSN 1053-587X. doi: 10.1109/tsp.2006.881199. URL `http://dx.doi.org/10.1109/tsp.2006.881199`.

[2] F. J. Anscombe. The transformation of poisson, binomial and negative-binomial data. *Biometrika*, 35(3/4):pp. 246–254, 1948. ISSN 00063444. URL `http://www.jstor.org/stable/2332343`.

[3] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. Non-Local Means Denoising. *Image Processing On Line*, 2011, 2011. doi: 10.5201/ipol.2011.bcm_nlm.

[4] Yang Cao, Yupin Luo, and Shiyuan Yang. Image denoising with gaussian mixture model. In *Image and Signal Processing, 2008. CISP '08. Congress on*, volume 3, pages 339–343, 2008. doi: 10.1109/CISP.2008.312.

[5] T. Chan, S. Esedoglu, F. Park, and A. Yip. Recent developments in total variation image restoration. In *In Mathematical Models of Computer Vision*. Springer Verlag, 2005.

[6] S.G. Chang, Bin Yu, and M. Vetterli. Adaptive wavelet thresholding for image denoising and compression. *Image Processing, IEEE Transactions on*, 9(9):1532–1546, 2000. ISSN 1057-7149. doi: 10.1109/83.862633.

[7] Joseph Salmon Charles-Alban Deledalle and Arnak Dalalyan. Image denoising with patch based pca: local versus global. In *Proc. BMVC*, pages 25.1–25.10, 2011. ISBN 1-901725-43-X. http://dx.doi.org/10.5244/C.25.25.

[8] P. Chatterjee and P. Milanfar. Is denoising dead? *Image Processing, IEEE Transactions on*, 19(4): 895–911, 2010. ISSN 1057-7149. doi: 10.1109/TIP.2009.2037087.

[9] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3D transform-domain collaborative filtering. *IEEE Transactions on image processing*, 16(8), August 2007.

[10] Mario A T Figueiredo and A.K. Jain. Unsupervised learning of finite mixture models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(3):381–396, 2002. ISSN 0162-8828. doi: 10.1109/34.990138.

[11] Samuel W. Hasinoff. Photon, poisson noise. Encyclopedia of Computer Vision, K. Ikeuchi and R. Kawakami, eds. Springer Science+Business Media, 2012. To appear.

[12] John Immerkaer. Fast noise variance estimation. *Computer Vision and Image Understanding*, 64(2):300 – 302, 1996. ISSN 1077-3142. doi: http://dx.doi.org/10.1006/cviu.1996.0060. URL http://www.sciencedirect.com/science/article/pii/S1077314296900600.

[13] S. M. Kay. *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice Hall, Englewood Cliffs, NJ, 1993.

[14] M. Lebrun, Antoni Buades, and Jean-Michel Morel. A nonlocal bayesian image denoising algorithm. *SIAM J. Imaging Sciences*, 6(3):1665–1688, 2013.

[15] Marc Lebrun. An analysis and implementation of the BM3D image denoising method. Image Processing On line, August 2012. http://dx.doi.org/10.5201/ipol.2012.l-bm3d.

[16] Marc Lebrun and Arthur Leclaire. An Implementation and Detailed Analysis of the K-SVD Image Denoising Algorithm. *Image Processing On Line*, 2012, 2012. doi: 10.5201/ipol.2012.llm-ksvd.

[17] Ce Liu, R. Szeliski, Sing Bing Kang, C.L. Zitnick, and W.T. Freeman. Automatic estimation and removal of noise from a single image. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(2):299–314, 2008. ISSN 0162-8828. doi: 10.1109/TPAMI.2007.1176.

[18] P. Milanfar. A tour of modern image filtering: New insights and methods, both practical and theoretical. *Signal Processing Magazine, IEEE*, 30(1):106–128, 2013. ISSN 1053-5888. doi: 10.1109/MSP.2011.2179329.

[19] K. B. Petersen and M. S. Pedersen. The matrix cookbook, nov 2012. URL http://www2.imm.dtu.dk/pubdb/p.php?3274. Version 20121115.

[20] Javier Portilla, Vasily Strela, Martin J. Wainwright, and Eero P. Simoncelli. Image denoising using scale mixtures of gaussians in the wavelet domain. *IEEE Trans. Image Process*, 12:1338–1351, 2003.

[21] R. Rubinstein, T. Peleg, and M. Elad. Analysis k-svd: A dictionary-learning algorithm for the analysis sparse model. *Signal Processing, IEEE Transactions on*, 61(3):661–677, 2013. ISSN 1053-587X. doi: 10.1109/TSP.2012.2226445.

[22] Shen-Chuan Tai and Shih-Ming Yang. A fast method for image noise estimation using laplacian operator and adaptive edge detection. In *Communications, Control and Signal Processing, 2008. ISCCSP 2008. 3rd International Symposium on*, pages 1077–1081, 2008. doi: 10.1109/ISCCSP.2008.4537384.