Texture and edge adaptive Weak Matching Pursuit

Cristian Canton Ferrer
Project Assistant: **Rosa M. Figueras i Ventura**
Professor: **Pierre Vandergheynst**
Signal Processing Institute,
Swiss Federal Institute of Technology (EPFL)

August 10, 2003

Esquemes Híbrids de Matching Pursuit per codificació d'image

Cristian Cantón Ferrer
Projecte Final de Carrera
Project Assistant: Rosa M. Figueras i Ventura
Director: Dr.Pierre Vandergheynst

Lausanne, 11 d'Agost del 2003
LTS, École Polytechnique Fédérale de Lausanne
Suïssa

En el desenvolupament de l'actual món tecnològic que ens envolta, el paper de les imatges digitals és vital. La optimització del processos relacionats amb aquest àmbit en termes de captura, emmagatzemament i transmissió representa un dels reptes més desafiants del processament d'imatge. Tota màquina capaç de rebre i enviar dades està incorporant la possibilitat de treballar també amb imatges i aquesta tendència sembla que hagi de ser més important en el futur. El fet de voler transmetre i rebre imatges amb qualsevol tipus d'aparell i en molts entorns diferents fa que s'hagi d'adaptar la representació d'aquest tipus de dades a les possibilitats del canal i del receptor: per a un canal de banda estreta cal que aquesta representació sigui més compacta.

El principal problema de l'emmagatzemament d'una imatge és l'elevat nombre de dades necessaries per representar-les. Per tal de reduir aquesta quantitat de dades s'han desenvolupat moltes tècniques diferents: representacions polinòmiques, Wavelets, DCT,... Tot i així el decreixement de l'error que la codificació amb un nombre finit de coeficients dóna, quan parlem d'imatges (senyals en dues dimensions) no és mai tan bona com quan parlem de senyals en una sola dimensió. Donat que en condicions adverses el nombre de bits que podem emprar per representar el senyal ve limitat per les característiques del canal o del receptor, el fet de tenir el decreixement de l'error major possible és vital per a millor la qualitat de la representació. Per aquest motiu, noves tècniques de codificació han estat estudiades. Aquestes tècniques han estat dissenyades no per treballar amb alts bit rates, on standards com el JPEG2000 són suficientment bons, però si per baixos bit rates on sempre hi ha una pèrdua de qualitat. Aquesta pèrdua de qualitat que les tècniques standard introdueixen quan representant imatges amb poques dades fan que el resultat tingui una mala qualitat visual.

L'algoritme Matching Pursuit a donat bons resultats quan s'intenta codificar una imatge amb pocs coeficients. Les seves propietats i la possibilitat d'obtenir representacions compactes el fan un bon candidat per un futur sistema de codificació d'imatges. Però el principal problema d'aquest algoritme (aplicat en la seva versió original per codificar imatges) és que la descomposició obtinguda s'obté tractant la imatge com un conjunt de píxels

sense cap sentit semàntic, és a dir no es té en compte el contingut de la imatge. Llavors, tenir en compte la forma en com els éssers human percevem les informació visual durant el procés de codificació pot proporcionar millors representacions en termes de qualitat visual. Dos nous mètodes són presesentats en aquest projecte que acompleixen el nostre objectiu. El primer és basa en l'ús d'una sèrie de màsqueres de probabilitat que donen més pes a les zones on hi ha concentrada més informació visual. Aquestes màsqueres seran dues (en el nostre model). La primera ressaltarà les àrees amb els contorns més importants (on hi sol haver més informació) i la seva definició es basa en una mesura de contrast sobre la imatge. De fet, la percepció humana està basada en mesures de contrast i, per tant, aquesta màscara acompleix la nostra perspectiva de ser fidels al sistema visual humà. La segona màscara detectarà i ressaltarà les textures presents a la imatge, basant-se en el mètode de E.Simmoncelli, que també es basa en la percepció humana de les textures. L'ús d'aquestes màsqueres en el procés de codificació no afecta el decreixement exponencial dels coeficients (propietat ja inherent al Matching Pursuit) cosa que permet una quantització eficient que resulta en un bit rate reduït.

El segon mètode proposat en aquest projecte es basa en una reordenació ("scrambling") de la descomposició donada per l'algoritme standard Matching Pursuit per tal d'aconseguir una representació on els termes que continguin més informació visual es representin en primer lloc. El criteri per dur a terme aquesta reordenació es basa a la seva vegada en les màsqueres de probabilitat anteriorment citades. Les aventatges d'aquests dos mètodes fins aquí citats són que les descomposicions obtingudes, tot i tenir un PSNR inferior a la descomposició generada per l'algoritme standard Matching Pursuit, la seva qualitat visual és millor. Això és una característica desitjable quan tractem amb bit rates molt baixos: mesurar la qualitat d'una imatge per com es veu en comptes del seu PSNR (que no es una bona referència visual d'una imatge).

Finalment, una nova versió de l'algoritme original Matching Pursuit: dur a terme la descomposició per sub-zones. En altres paraules, definir una partició de l'imatge original en zones amb una característica homogènia, textures o contorns, i aplicar una descomposició local, dissenyant diccionaris adaptats a les característiques d'aquella partició. El resultat d'aplicar aquest procés comporta una reducció del 60% -80% del temps per a dur a terme la codificació.

Als meus pares

Acknowledgements

En primer lloc donar les gràcies molt especialment als meus pares que van animar-me en tot moment a venir a Suïssa i em van donar i em donen tot el suport que necessitava i més. Sense ells això no hagués estat possible. Gràcies papa i mama.

També gràcies a la meva família en general: al meu germà, els meus avis, tiets, cosí i cosines que sempre van estar pendents de mi en tot moment i van fer que la distància no fos un motiu de nostàlgia.

A la Rosa Maria, la meva simpàtica "assistant" durant aquests sis mesos a l'EPFL. Per el seu guiatge durant tot el projecte i pels bons moments que hem passat junts. Pel seu bon humor, la seva constància, la seva disponibilitat a totes hores. Gracies Rosa!

My most grateful thanks to Professor Pierre Vandergheynst who did make possible my stay at EPFL and in Switzerland (with the kind economical help of LTS). Also, thanks for his helpful ideas and advices during the time I did my thesis.

Grazie ai miei amici italiani Alessandro, Emilio e Gianluca [39] per tutti i momenti indimenticabili che noi abbiamo passato insieme. E grazie per tutte le lezioni d'italiano maccherronico e per avermi insegnato come si cucina la vera pasta. Anche con scoppolamento a destra e la supercazzola prematurata, ma sempre come si fossi antani!E senza fuochi fatui!!

Donar les gràcies a tota la gent que he conegut a Suïssa és difícil perqué he tingut la sort de conèixer-ne molts. Primer gràcies als meus companys de laboratori: Marc, Àlvaro, Irene, Vanessa, David, Gianluca, Emilio, Alessandro, Mei, Yin, Julien, Isa,... i als companys EPFL-ians: Jordi (quins bons moments hem passat junts!), Eva, Pablo, Robert, Javi, Jaime, Ana, Santi, Pedro, Rodrigo, Teresa, i tants d'altres.

A toda la gente que se ha cruzado en mi camino en la residencia: Patricia, Amaya, Bea, las Marias, Elisabet, Prado, il pianista Salvatore, Gisela, Dani, Magda,...

Un saludo también muy especial para aquellos que me acogieron desde mi primer día en Lausanne y me hicieron sentir un clima siempre hogareño: María y Jesús Bustamante. Gracias por vuestros ánimos y por los momentos que hemos pasado juntos.

A tots els meus amics de Barcelona, molt especialment a l'Ester, i tots els altres que no cito per no omplir pàgines i pàgines!

También un recuerdo para mi amiga Virginia i todos mis companyeros de viaje alrededor del mundo.

Al meu gos Beto que, desde el cel del gossos, deu estar bordant-me d'alegria perqué torno a casa.

És realment molt difícil enumerar tota aquella gent que m'ha ajudat a arribar on sóc ara, a tots aquells que no cito, també els hi dono les gràcies!

Finalment, també donar les gràcies als serveis de Relacions Internacionals de l'UPC per haver-me donat l'oportunitat de venir a Suïssa.

Cristian Cantón Ferrer

Contents

List of Figures

Abstract

The objective of image coding is to reduce the number of bits needed to represent an image, while making as few as possible perceptual distortion to the image. Images coded at low bit-rates, say below 0.5bpp, bear the loss of details and sharpness, as well as various artifacts which are perceptually objectionable. On the other hand, with the need for transmission and storage of more and larger images, the demand for higher compression is also increasing. When the transmission is restricted by a narrow bandwidth channel or the storage is constrained to spend as few bytes as possible, we must adapt the image representation to fit such scenarios. The existent techniques, when allowing a low number of bits, introduce distortions to the image that are very unpleasant to the observer, and so a simplification of the sent image or a coding method that automatically does it is needed in order to have an acceptable image representation.

In previous works [31], Matching Pursuit was presented as a powerful technique to code images for very low bit-rate channels. In this thesis, three dictionaries have been built based on the following atoms: Anisotropic Refinement (suitable to code edges), Gabor (to code textures) and Gauss (to code the image baseband). Furthermore, three new coding schemes have been proposed with the aim to embed the issue of visual quality into the coding process. **Hybrid Matching Pursuit**, that performs the decomposition taking into account the most meaningful areas of the image, **Scrambled Matching Pursuit** that computes a "visually scaled" output stream based on the decomposition given by the standard Matching Pursuit and **Split Matching Pursuit** that splits the original image into sub-images and performs Matching Pursuit into each sub-image; this method speeds-up the standard algorithm around 60-80%.

These coders, when working with a low amount of coefficients, have good performance and give better results than other conventional techniques such as DCT or wavelets. The main advantages regarded by these schemes are its adequacy to code properly the most meaningful areas of an image, its robustness to quantization, scalability and simplicity of the decoder.

Chapter 1

Introduction

1.1 Where are we?

Optimizing the performance of digital imaging systems with respect to the capture, display, storage and transmission of visual information represents one of the biggest challenges in the field of image and video processing.

One of the main problems of images is the big amount of data we need to have them fully represented. A large number of techniques to reduce this amount of data have been implemented in order to be able to transmit digital images with low bit rates. These techniques go from very simple ones (like polynomial representations) to more elaborated ones (fractal coding, DCT,...). Yet the coding error decay (the decay of the module of the error due to the representation when adding a coefficient) in image (2D signals) is never as good as in 1D signals. And when the number of bits "allowed" to represent a signal is limited for the characteristics of the channel or the receiver, the fact of maximizing the error decay is vital.

That is the reason why new coding techniques have been introduced. Not for large bit rates, where standards like JPEG2000 are good enough, but for smaller bit rates, where there is always a loss of quality. The loss of quality that the actual coding techniques introduce when coding images at extremely low bit rates leads to not visually pleasant representations.

Matching Pursuit algorithm has showed good results when coding images with few coefficients. Its properties and the possibility to get sparse representation of images (that is with very few coefficients) make it a good direction where to work on. But the standard flavor of Matching Pursuit is completely based on the image as a meaningless set of pixels without taking in account any relevant aspect of it, in terms of what represents in the image itself. Taking into account the way humans perceive visual information can be greatly beneficial for this task. If this visual information can be taken into account when effectuating the coding process, the resulting representation would be more pleasant to the observer.

In this dissertation, three novel schemes based on the former Matching Pursuit tech-

nique are presented in order to improve the coding results in relation with the perception of the reconstructed image. Furthermore, improvements over the existing Matching Pursuit results have been done by adding new dictionaries to code determinate features (like textures) of the image. Maybe this new technique is the gate to the future of image coding.

1.2 Visual quality

Matching Pursuit works out by choosing iteratively the coefficients with largest energy. In some cases, this leads to achieve sparse representations of the input signal, very suitable for compression. It does not pay attention to the areas that contain more visual information in the image, so that is the strongest contours and patterns. Hence, we would try to develop a variant of this algorithm in order to enhance the most meaningful areas of the image and create a perceptually "nice" representation. A way to do this is to use a set of probability masks that detect this relevant areas and embed this information into the coding process in such a way more atoms are placed in these interest areas.

These probability masks will be two. The first one will be based on contrast measures (i.e. the perception of stimuli in relation to their surround) to detect where are the strongest edges of the image; in fact, human perception is based on the measures of contrast, hence, this mask fits our aim to remain close to the human visual system. The other mask will detect the textures present in the image by a method [84] also based on the responses of the early visual cortex cells [72]. Hence, it seems to be a good starting point to develop coding techniques related with the perception of the images by the humans.

1.3 Organization

The order given to this dissertation pretends to make it easy to understand Matching Pursuit, the new schemes based on it and the reasons that have brought to the study of these techniques as well. The work is organized as follows:

Chapter 2 includes the Signal Processing basics. First of all we have a short introduction to the Fourier Transform and the Discrete Fourier Transform. Afterwards, there is a short review on edge detectors (basically, linear kernels) and an introduction to texture detectors. Finally, a short contraposition between pixel-based metrics and subjective metrics is written. Furthermore, here we show the inviability of pixel-based metrics such PSNR or MSE to evaluate the quality of an image giving the example of masking effect.

Chapter 3 makes an introduction to Adaptive Greedy Approximations of signals. This approximation problem belongs to the set of NP-hard problems, hence the optimal approximation is not computationally affordable. Instead, sub-optimal approximations have been purposed, among them Matching Pursuit. A fully presentation of this algorithm is done and an optimization based on a Full Search in the frequency

domain as well. Finally, Weak Matching Pursuit is introduced and the computer implementation we did to perform our decompositions (based on distributed programming).

Chapter 4 gives a general description of the three dictionaries used by our Hybrid Matching Pursuit coders. A thorough mathematical study over its properties and design is done.

Chapter 5 shows the architectures and properties of the three new Hybrid Matching Pursuit coding schemes. With this descriptions there are the main results obtained: their performances, the quality achieved, the quantization and the codification.

Chapter 6 announces the main conclusions we can get from this work, as well as some future work.

Part I

Theory and Background

Chapter 2

Signal Processing Background

2.1 Introduction

This chapter pretends to be a brief introduction to the concepts and theory necessary to have a full understanding of the work presented in this thesis. It does not pretend to be a full extensive explanation of all this concepts but a short review of them. Actually, a person familiar with Signal Processing could skip this chapter due to its simplicity.

In the last pages of this thesis you can find a wide bibliography that could help you to understand thoroughly the concepts presented here. More specific and recommended bibliography to a fully understand of this work would be:

- For an introduction to the basis of Image Processing, the following references are recommended [9],[49].
- Mallat's book [63] for a complete description on Fourier spaces, wavelets and a brief theoretical approach to Matching Pursuit.
- Matching Pursuit is quite fairly explained in the paper by Mallat and Zhang [62]. Despite it is explained for mono-dimensional signals, the principle is extensible to multidimensional signals.
- For a short review on edge and texture detection, the following references are most interesting [65],[66].

2.2 Fourier Kingdom

The Fourier transform expresses a signal as the sum of a series of complex exponentials (sines and cosines). Due to the fact that the sinusoidal waves are eigenvectors of linear time-invariant/space-invariant operations, the Fourier transform is appropriate for linear time-invariant/space-invariant signal processing. The properties of the Fourier Transform

make it very useful for signal processing of continuous or discrete signals. Mainly, continuous signals belong to the analogical signal processing field, hence the Continuous Fourier Transform is defined for this type of signals. In image processing, the information is represented by a discrete set (an image) formed by $M \times N$ values (pixels) which hold a value (continuous or discrete if they are quantized). In this way, the Discrete Fourier Transform is a suitable tool for representing and manipulating the image information. First, the Fourier Transform will be introduced in the continuous time domain and then it will be extrapolated to the Discrete Fourier Transform, in non-continuous (so, discrete) time.

2.2.1 Continuous Fourier Transform in 2D

The Continuous Fourier Transform for two dimensional signals is defined as a linear operator $\mathcal{F} : \mathbb{R} \times \mathbb{R} \to \mathbb{C} \times \mathbb{C}$ as follows:

$$F(\omega_x, \omega_y) = \mathcal{F}[f(x,y)](\omega_x, \omega_y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\, e^{-j(x\omega_x + y\omega_y)} dx\, dy, \tag{2.1}$$

where $e^{j(x\omega_x + y\omega_y)}$, when written in polar coordinates, gives the expression of a plain wave:

$$e^{j(x\omega_x + y\omega_y)} = e^{j\rho(x\cos\theta + y\sin\theta)}, \tag{2.2}$$

with $\rho = \sqrt{\omega_x^2 + \omega_y^2}$. This wave propagates in the direction of θ and oscillates at the frequency ρ.

The Fourier Transform of $f(x,y)$ is then the amplitude of each sinusoidal wave $e^{j(x\omega_x + y\omega_y)}$, and it represents the influence of each frequency in the signal.

The Inverse Fourier Transform is given by the following integral:

$$f(x,y) = \mathcal{F}^{-1}[F(\omega_x, \omega_y)](x,y) = \frac{1}{4\pi^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(\omega_x, \omega_y)\, e^{j(\omega_x x + \omega_y y)} d\omega_x\, d\omega_y. \tag{2.3}$$

As the FT gives a full representation of the signal, it is an invertible operation and supposes no loss of information.

2.2.2 Propertied of the Continuous Fourier Transform in 2D

The Fourier Transform has certain properties that make it very useful and appropriate for signal processing. Here there are presented the main properties without a fully demonstration of them (mainly, their demonstrations are directly found by applying the Fourier Transform definition).

Linearity Due to the fact that the integration is a linear operator, the Fourier Transform has also the properties of linear operators: commutation with addition and product by a scalar. Hence:

$$\mathcal{F}[\lambda f(x,y) + \mu g(x,y)] = \lambda F(\omega_x, \omega_y) + \mu G(\omega_x, \omega_y). \tag{2.4}$$

Transposition This property shows that the Fourier Transform of the transpose of $f(x, y)$, being the transpose $f(-x, -y)$, is the transpose of the Fourier Transform:

$$\mathcal{F}[f(-x, -y)] = F(-\omega_x, -\omega_y). \tag{2.5}$$

Conjugation The conjugation property says that the Fourier Transform of the conjugate of a signal is the conjugated and transposed Fourier Transform of this signal:

$$F[f^*(x, y)] = F^*(-\omega_x, -\omega_y), \tag{2.6}$$

where $*$ represents the complex conjugate of the function.

Scaling Scaling the function is the same as multiplying the variables x and y by a constant $s_x \neq 0$ and $s_y \neq 0$ respectively. We find that scaling in space is equivalent to the inverse of the scaling in frequency (with a multiplying factor):

$$\mathcal{F}[f(s_x x, s_y y)] = \frac{1}{|s_x s_y|} F\left(\frac{\omega_x}{s_x}, \frac{\omega_y}{s_y}\right). \tag{2.7}$$

Translation This property asserts that a translation in space is equivalent to a modulation in the frequency domain:

$$\mathcal{F}[f(x - p_x, y - p_y)] = e^{j(\omega_x p_x + \omega_y p_y)} F(\omega_x, \omega_y). \tag{2.8}$$

Modulation This is the inverse property of translation: the Fourier Transform of a modulated signal is a frequency translated signal.

$$\mathcal{F}[e^{j\left(\omega_{x_0} x + \omega_{y_0} y\right)} f(x, y)] = F(\omega_x - \omega_{x_0}, \omega_y - \omega_{y_0}). \tag{2.9}$$

Derivation This property says that partial derivation with respect to one spatial variable in the space domain is equivalent to the non-derived transformed function multiplied by its respective frequency variable in the frequency domain:

$$\mathcal{F}\left[\frac{\partial^m f(x, y)}{\partial x^m}\right] = (j\omega_x)^m F(x, y). \tag{2.10}$$

If we derive with respect to y instead of x in the frequency domain we will multiply by ω_y instead of ω_x.

Deriving in the frequency domain is equivalent to deriving in the spatial domain, with a change of sign, so we can also deduce the dual property for derivation in the frequency domain:

$$\mathcal{F}[(-jx)^m f(x, y)] = F^{(m)}(\omega_x, \omega_y). \tag{2.11}$$

Rotation A rotation of θ in $f(x, y)$ causes the same rotation in the transformed domain $F(\omega_x, \omega_y)$:

$$\mathcal{F}[f(x,y)\circlearrowleft\theta] = F(\omega_x, \omega_y)\circlearrowleft\theta. \tag{2.12}$$

Convolution To perform a linear convolution in the space domain is equivalent to a product in the Fourier domain:

$$\mathcal{F}[f(x,y) * g(x,y)] = F(\omega_x, \omega_y) \cdot G(\omega_x, \omega_y). \tag{2.13}$$

The property is very important and the reader should recall it because in future chapters will be used thoroughly. Note the save of computational load between performing a convolution in space domain and a product in the frequency domain.

Separability Due to integral properties, the Fourier Transform has the separability condition. That is equivalent to say that the 2D Fourier Transform of a function is equal to a product of 1D Fourier Transform along the x and y directions:

$$\mathcal{F}_{\text{2D}}[\text{Image}] = \mathcal{F}_{\text{1D}}[\text{Rows}]\left\{\mathcal{F}_{\text{1D}}[\text{Columns}]\right\}. \tag{2.14}$$

This property is useful for simplifying the calculation of the Fourier Transform when implementing it.

Information in images Most of the information contained in images comes from the contours and edges[1] [30]. In the Fourier domain, the contours are mainly represented by the phase of the Fourier transform. This implies that most of the information of the images is in the phase, not in the module, as can be seen in the example depicted in Figure 2.1.

2.2.3 Discrete Fourier Transform

The 2D Discrete Fourier Transform for a $M \times N$ pixels image is defined as follows

$$X[k,l] \triangleq \frac{1}{\sqrt{MN}} \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} x[m,n] e^{-j2\pi\left(\frac{mk}{M}+\frac{nl}{N}\right)} \qquad 0 \le k \le M, 0 \le l \le N. \tag{2.15}$$

And the inverse transform is:

$$x[m,n] \triangleq \frac{1}{\sqrt{MN}} \sum_{k=0}^{M-1}\sum_{l=0}^{N-1} X[k,l] e^{j2\pi\left(\frac{mk}{M}+\frac{nl}{N}\right)} \qquad 0 \le m \le M, 0 \le n \le N, \tag{2.16}$$

where $1/\sqrt{MN}$ is a normalising factor to have a unitary transformation.

[1] Realize that we are talking in term of *visual information* instead of PSNR or other pixel based measures.

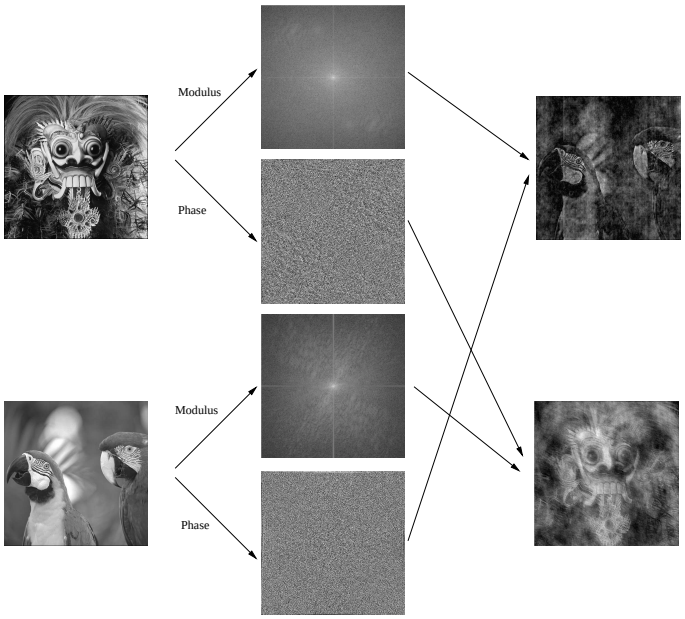


Figure 2.1: Information in images. Demonstration that the greatest part of the information is regarded by the phase of the Fourier Transform instead of the modulus.

The Discrete Fourier Transform in 2D is narrowly connected with the Fourier Transform in the sense that the first can be calculated as a discretisation of the seconed. That is, in other words, the DFT is a sampled version[2] of the FT:

$$X[k,l] = X\left(e^{j\omega_x}, e^{j\omega_y}\right)|_{\omega_x=\frac{2\pi}{M}k,\omega_y=\frac{2\pi}{N}l}, \tag{2.17}$$

where the size of the image is $M \times N$.

The DFT has the same properties as FT, but it has also to them the periodicity and the symmetry characteristics. A transformed DFT signal, according to the Nyquist theorem, is periodised as follows:

$$X[k,l] = X[k + r_1 M, l + r_2 N], \tag{2.18}$$

with $r_1, r_2 \in \mathbb{N}$. Moreover, the DFT has the following symmetry respost to the zero frequency:

$$X[k,l] = X^*[M-k, N-l], \tag{2.19}$$

where $*$ representes the complex conjugated.

[2]Usually sampled uniformly over the time. Non-uniform sampling is explained thoroughly in [2].

2.2.4 Fast Fourier Transform (FFT)

When the work of Cooley and Tukey [16][3] appeared in 1965, signal processing field suffered a revolution: Fast Fourier Transform was born. FFT is an optimization of the calculus of the DFT in the sense that for vectors (or matrices) with a dimension 2^n with $n \in \mathbb{N}$, the computational load was reduced from N^2, in the DFT case for a square image of N pixels, to $N \log_2 N$, in the case of FFT. In Image Processing applications, FFT is commonly used and it will be the basis of some optimizations in further chapters.

2.3 Edge detection filters

Edge detection is a fundamental part of this project. In future chapters, a crucial part of our Hybrid Matching Pursuit coder will take use of an edge detection filter. Hence, a short review of the standard edge detection filters is done here as well as few remarks of comparison. This does not pretend to be an extensive explanation, just few words; for more information review [49], [9].

2.3.1 First-Derivative methods

Most edge-detecting operators can be seen as gradient-calculators, so, that is, based on the first-derivative. Recall the definition of the gradient operator ∇:

$$\nabla = \begin{bmatrix} \frac{\partial}{\partial x} \\ \\ \frac{\partial}{\partial y} \\ \cdot \end{bmatrix} \tag{2.20}$$

Many of these detectors are based in some way on measuring the intensity gradient at a point in the image. Because the gradient is a continuous-function concept and we have discrete functions (images), we have to approximate it. Since derivates are linear and shift-invariant, gradient calculation is most often done using convolution. Numerous linear kernels have been proposed for finding edges based on the calculus of the first derivative and some of them are presented here.

Roberts Kernels Since an edge detector is looking for differences between pixels, one way to find edges is to use an operator that calculates $I(\bar{x}_i) - I(\bar{x}_j)$ for two pixels i and j in a neighborhood. Matematically, these are called *forward differences*:

$$\frac{\partial I}{\partial x} \approx I(x+1, y) - I(x, y). \tag{2.21}$$

[3] In fact, Fast Fourier Transform was not new. Gauss had already published a work that was a primitive form of the modern FFT [48].

The Roberts kernels attempt to implement this using the following kernels:

$$g_1 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \qquad g_2 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, \tag{2.22}$$

While these are not specifically derivatives with respect to x and y, they are derivatives with respect to the two diagonal directions. These can be thought of as components of the gradient in such a coordinate system. So, we can calculate the gradient magnitude by calculating the length of the gradient vector:

$$g = \sqrt{(g_1 * I)^2 + (g_2 * I)^2}, \tag{2.23}$$

where $*$ is the convolution operator. The main drawback of this edge detector is that Roberts kernels are in practice too small to reliably find edges in the presence of noise.

Prewitt Kernels The Prewitt kernels are based on the idea of the *central difference*:

$$\frac{\partial I}{\partial x} \approx \frac{I(x+1, y) - I(x-1, y)}{2}. \tag{2.24}$$

This corresponds to the following convolution kernel:

$$\frac{1}{2} \begin{pmatrix} -1 & 0 & 1 \end{pmatrix}. \tag{2.25}$$

By rotating this kernel 90 degrees, we get $\partial I/\partial y$.

These kernels are, however, sensitive to noise; a method to reduce some of the effects of noise, we can perform *averaging*. This is done in the Prewitt kernels by averaging in y when calculating $\partial I/\partial x$ and by averaging in x when calculating $\partial I/\partial x$. These produce the following kernels:

$$\frac{\partial}{\partial x} = \frac{1}{6} \begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix} \qquad \frac{\partial}{\partial y} = \frac{1}{6} \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{pmatrix}, \tag{2.26}$$

Together, these kernels give us the components of the gradient vector.

Sobel Kernels The Sobel kernels also rely on central differences, but give greater weight to the central pixels when averaging

$$\frac{\partial}{\partial x} = \frac{1}{8} \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \qquad \frac{\partial}{\partial y} = \frac{1}{8} \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}, \tag{2.27}$$

Together, these also give us the components of the gradient vector.

The Sobel kernels can also be thought as 3×3 approximations to first-derivative of Gaussian kernels. That is, it is equivalent to first blurring the image using a 3×3 approximation of the Gaussian and then calculating first derivates. This is because convolution (and derivatives) are commutative and associative:

$$\frac{\partial}{\partial x}(I * G) = I * \frac{\partial}{\partial x} G. \tag{2.28}$$

2.3.2 Second-Derivative Methods

Most edges are, however, not sharp dropoffs. They are often gradual transitions from one intensity to another. What usually happens in this case is that you get a rising gradient magnitude, a peak of the gradient magnitude, and then a falling gradient magnitude. Finding optimal edges (maxima of gradient magnitude) is thus equivalent to finding places where the second derivative is zero. For two dimensions, there is a single measure, similar to the gradient magnitude that measures second derivatives, the Laplacian operator ∇^2:

$$\nabla^2 = \nabla \cdot \nabla = \begin{bmatrix} \frac{\partial}{\partial x} \\ \\ \frac{\partial}{\partial y} \end{bmatrix} \cdot \begin{bmatrix} \frac{\partial}{\partial x} \\ \\ \frac{\partial}{\partial y} \end{bmatrix} = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}. \tag{2.29}$$

Laplacian The Laplacian of the Gaussian method finds edges by looking for zero crossings after filtering an image with a Laplacian of Gaussian filter (this filter actually calculate the second derivative of an image). The finite differences approximation for the second derivative is:

$$\begin{aligned} \frac{\partial^2 I(x,y)}{\partial x^2} &\approx [I(x+1,y) - I(x,y)] - [I(x,y) - I(x-1,y)] && (2.30) \\ &= I(x+1,y) - 2I(x,y) + I(x-1,y) && (2.31) \end{aligned}$$

Finally, the convolution kernel for this method leads to:

$$\begin{pmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{pmatrix}. \tag{2.32}$$

The drawback of this method is very susceptible to noise. This sensitivity comes not just from the sensitivity of the zero-crossings, but also of the differentiation. In general, the hight the derivative, the more sensitive the operator.

All these edge detector filters lead to interesting results but none of them gives enough resolution and accuracy to be embedded in our system. The solution is provided by two more advanced techniques of edge detection: Canny edge detector and contrast-based edge detector.

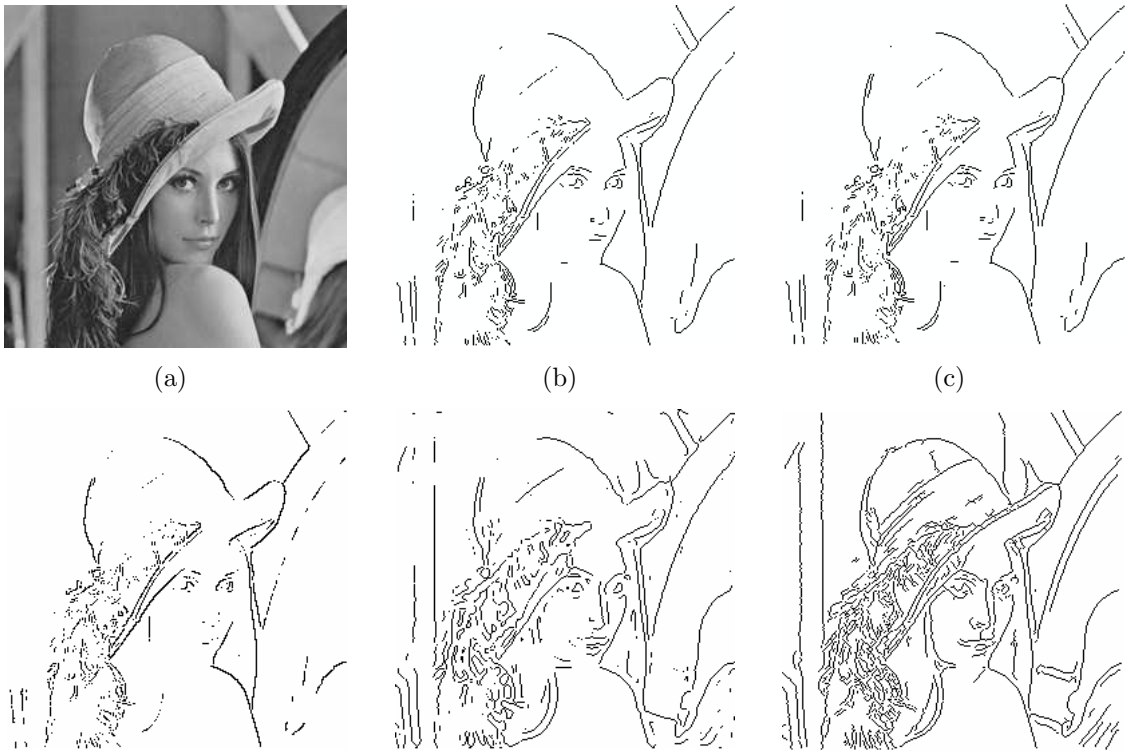


Figure 2.2: Edge detectors comparison. (a) Original Lena image, (b) Sobel detector, (c) Prewitt detector, (d) Roberts detector, (e) Laplacian detector, (f) Canny detector. To obtain the results shown here, the estimation of the first derivative has been thresholded in order to facilitate the comparison between them.

2.3.3 Canny Edge Detector

Among all the existing edge detectors, Canny's one (proposed in [8]) is known to many as the optimal edge detector. Canny's intentions were to enhance the many edge detectors already proposed already by taking into account a list of criteria:

- The first, and most obvious, is low error rate; it is important that edges occurring in images should not be missed and that there should be no responses to non-edges.
- The second criterion is that the edge points must be well localized, in other words, the distance between the edge pixels found by the detector and the actual edge has to be minimum.
- A third criterion is to have only one response to a single edge. This was implemented because the first two points were not substantial enough to completely eliminate the possibility of multiple responses to an edge.

Based on this criteria, the Canny edge detector first smoothes the image to eliminate noise by convolving it with a Gaussian filter. Then finds the image gradient to highlight regions with high spatial derivatives. The algorithm then tracks along these regions and suppresses any pixel that is not at the maximum. The gradient array is now further reduced by hysteresis; hysteresis is used to track along the remaining pixels that have not been suppressed. This hysteresis uses two thresholds and if the magnitude is below the first threshold it is set to zero (made a nonedge). If the magnitude is above the high threshold, it is made an edge and if the magnitude is between the two thresholds, then it is set to zero unless there is a path from this pixel to a pixel with a gradient above.

In fact, this method gives better results than the other methods proposed in the former section. As depicted in Figure 2.2, Canny's edge detector detects many of the existing edges present in the image. But this is not enough for our purposes: an edge detector is required to be able to deal with all types of natural images, even the ill-posed ones, that is images with strong contrast or low luminance. In this case, Canny's edge detector gives poor results hence, other techniques are required. Furthermore, we would like to have an edge detector able to detect in the same way as the visual perception does: by detecting contrast differences.

2.3.4 Contrast measures

One of the main drawbacks of detecting edges in images is to deal with images that have high dynamic ranges. For example, let us examine a natural outdoor scene on a bright sunny day, one can easily observe huge luminance differences between areas in the shadow and in bright sunlight. The ratio can easily be in a range of 1 to 1000. When we try to apply the standard edge detection algorithms we notice that they detect the edges in the shadow areas weakly or event do not detect them. For our applications, it is necessary to have an edge detector that detects the edges locally not to mask the weak edges with the the strong ones. This property will be used into the creation of the probability masks in Chapter 3.

For our purposes, the contrast measure should exhibit a maximum at the edges locations, should decrease monotonically away from the edge and should not exhibit any zero crossing. This can simply be done by taking the derivative amplitude of the image, which can be computed by combining the derivatives along the lines and the columns of the image [46]. First, let us define a Gaussian pyramid to filter the different frequency bands of the original image. This pyramid is given by the convolution of the original image by the Gaussian filters defined as

$$\hat{\phi}_j(\omega_x, \omega_y) \triangleq \hat{\phi}(2^j\omega_x) \cdot \hat{\phi}(2^j\omega_y) \tag{2.33}$$

$$\hat{\phi}(\omega) = e^{-\frac{\omega^2}{2\sigma^2}}. \tag{2.34}$$

In order to avoid aliasing when sub-sampling the different bands of the pyramid, σ is

set to $\sigma = 0.45$. Then the derivative can be calculated:

$$\hat{D}_j^{(x)}(\omega_x, \omega_y) = j\omega_x \cdot \hat{\phi}_j(\omega_x, \omega_y) \cdot \hat{I}(\omega_x, \omega_y) \quad (2.35)$$

$$\hat{D}_j^{(y)}(\omega_x, \omega_y) = j\omega_y \cdot \hat{\phi}_j(\omega_x, \omega_y) \cdot \hat{I}(\omega_x, \omega_y) \quad (2.36)$$

$$D_j(x, y) = \sqrt{\left(D_j^{(x)}(x, y)\right)^2 + \left(D_j^{(y)}(x, y)\right)^2}, \quad (2.37)$$

where D_j is the amplitude of the derivative of the image in the j-th level of the multiresolution pyramid and $\hat{\phi}_j$ the low-pass filter (expressed in Fourier domain) used to get the j-th level of the multi-resolution pyramid from image I. If the cutoff of the low-pass filter is small enough compared to the Nyquist sampling rate, the derivative can be computed using a very simple difference operator:

$$D_j(x, y) \simeq 2^{j-1}\sqrt{(I(x+1, y) - I(x-1, y))^2 + (I(x, y+1) - I(x, y-1))^2}. \quad (2.38)$$

Finally, the contrast $C_j^D(x, y)$ is obtained by dividing the derivate by a low-pass image:

$$C_j^D(x, y) = \begin{cases} \frac{D_j(x,y)}{\phi_{j+1} * I(x,y)}, & D_j(x, y) > 0 \\ 0, & \text{else} \end{cases}. \quad (2.39)$$

2.4 Texture detection

2.4.1 Background

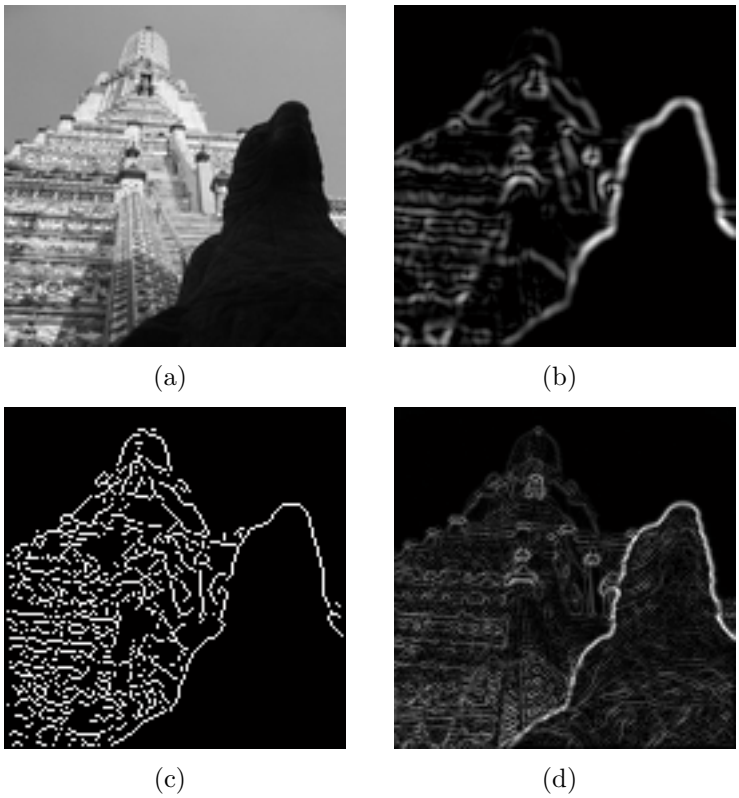
Although there is no strict definition of the image texture, it is easily perceived by humans and is believed to be a rich source of visual information, about the nature and three-dimensional shape of physical objects. Generally speaking, textures are complex visual patterns composed of entities, or subpatterns, that have a characteristic brightness, color, slope, size, etc. Thus texture can be regarded as a similarity grouping in an image [79]. The local subpattern properties give rise to the perceived lightness, uniformity, density, roughness, regularity, linearity, frequency, phase, directionality, coarseness, randomness, fineness, smoothness, granulation,etc., of the texture as a whole [59].

In texture analysis and detection there are two major issues:

- Feature extraction: to compute a characteristic of a digital image able to numerically describe its texture properties.
- Texture discrimination: to partition an image into regions each corresponding to a perceptually homogeneous texture.

Approaches to texture analysis are usually categorised into four different types:

- Structural.



(a) (b)

(c) (d)

Figure 2.3: High contrast images edge detection. In picture (a) there is the original image showing a high contrast between the Wat-Arun temple (Bangkok) at the back and the eagle in first plane. Picture (b) shows the inefficiency of the Gauss-Sobel estimation of the first derivative to detect the edges in the low contrasted areas (the eagle). In figure (c), threshold Canny's edge detector based on the results of (b) is unable to detect these weak, low contrasted, edges. Finally, in figure (d), contrast measure to detect the edges performs properly to detect the weak edges.

- Statistical.
- Model-based.
- Transform methods.

Structural approaches ([45], [59]) represent texture by well-defined primitives (called *microtextures*) and a hierarchy of spatial arrangements (*macrotextures*) of those primitives. To describe the texture, one must define the primitives and the placement rules. The choice of a primitive (from a set of primitives) and the probability of the chosen primitive to be placed at a particular location can be a function of location or the primitives near the location. The advantage of the structural approach is that it provides a good symbolic description of the image; however, this feature is more useful for synthesis than analysis

tasks. The abstract descriptions can be ill defined for natural textures because of the variability of both micro- and macrostructure and no clear distinction between them. A powerful tool for structural texture analysis is provided by mathematical morphology ([82], [13]). It has proved to be useful for bone image analysis, e.g. for the detection of changes in bone microstructure.

In contrast to structural methods, **statistical** approaches do not attempt to understand explicitly the hierarchical structure of the texture. Instead, they represent the texture indirectly by the non-deterministic properties that govern the distributions and relationships between the grey levels of an image. Methods based on second-order statistics (i.e. statistics given by pairs of pixels) have been shown to achieve higher discrimination rates than the power spectrum (transform-based) and structural methods ([93]). Human texture discrimination in terms of texture statistical properties is investigated in [54]. Accordingly, the textures in grey-level images are discriminated spontaneously only if they differ in second order moments. Equal second order moments, but different third-order moments require deliberate cognitive effort. This may be an indication that also for automatic processing, statistics up to the second order may be most important ([70]). The most popular second-order statistical features for texture analysis are derived from the so-called co-occurrence matrix ([45]).

Model based texture analysis ([17], [74], [12], [22], [64], [85]), using fractal and stochastic models, attempt to interpret an image texture by use of, respectively, generative image model and stochastic model. The parameters of the model are estimated and then used for image analysis. In practice, the computational complexity arising in the estimation of stochastic model parameters is the primary problem. The fractal model has been shown to be useful for modelling some natural textures. It can be used also for texture analysis and discrimination ([74], [11], [55]); however, it lacks orientation selectivity and is not suitable for describing local image structures.

Transform methods of texture analysis, such as Fourier [78], Gabor [20],[7] and wavelet transforms [61], [58] represent an image in a space whose co-ordinate system has an interpretation that is closely related to the characteristics of a texture (such as frequency or size). Methods based on the Fourier transform perform poorly in practice, due to its lack of spatial localization. Gabor filters provide means for better spatial localization; however, their usefulness is limited in practice because there is usually no single filter resolution at which one can localize a spatial structure in natural textures. Compared with the Gabor transform, the wavelet transforms feature several advantages:

- varying the spatial resolution allows it to represent textures at the most suitable scale,
- there is a wide range of choices for the wavelet function, so one is able to choose wavelets best suited for texture analysis in a specific application.

These properties make the wavelet transform attractive for texture segmentation. The problem with critically sampled wavelet transform is that it is not translation-invariant [5].

2.4.2 Simoncelli's Texture Detector

Among all these methods, statistical approach to texture analysis have been chosen due to its good performance on texture detection. There are few methods able to perform an analysis of an image to segment the textures but we have been chosen the method developed by E.Simoncelli and J.Portilla [84]. The model is parameterized by a set of statistical descriptors and the data given by them will allow to distinguish between a texture area or not. Furthermore, the choices of the set of statistical parameters have been made according to what is known for the Human Visual System (HVS). That is, how the textures are perceived by our eyes, so a good model for our purposes.

This texture detector is based on the decomposition of the original image I_0 over a steerable pyramid. This steerable pyramid is a linear multi-scaled, multi-oriented image decomposition. This representation is translation-invariant and rotation-invariant. The filters used to perform this steerable pyramid are inspired in the response of the neurons in the primary visual cortex to obtain a result closer to the human perception. For a detailed explanation over the construction of this pyramid, review [84] and [83].

Loosely speaking, Simoncelli's texture detector is based on the following steps:

1. Define the initial parameters, that is: number of scales (N_{sc}) and orientations (N_{or}) for the steerable pyramid and number of pixel neighbors (N_a).

2. Compute pixel statistics of the original input image I_0 (mean, variance, skewness, kurtosis, minimum and maximum).

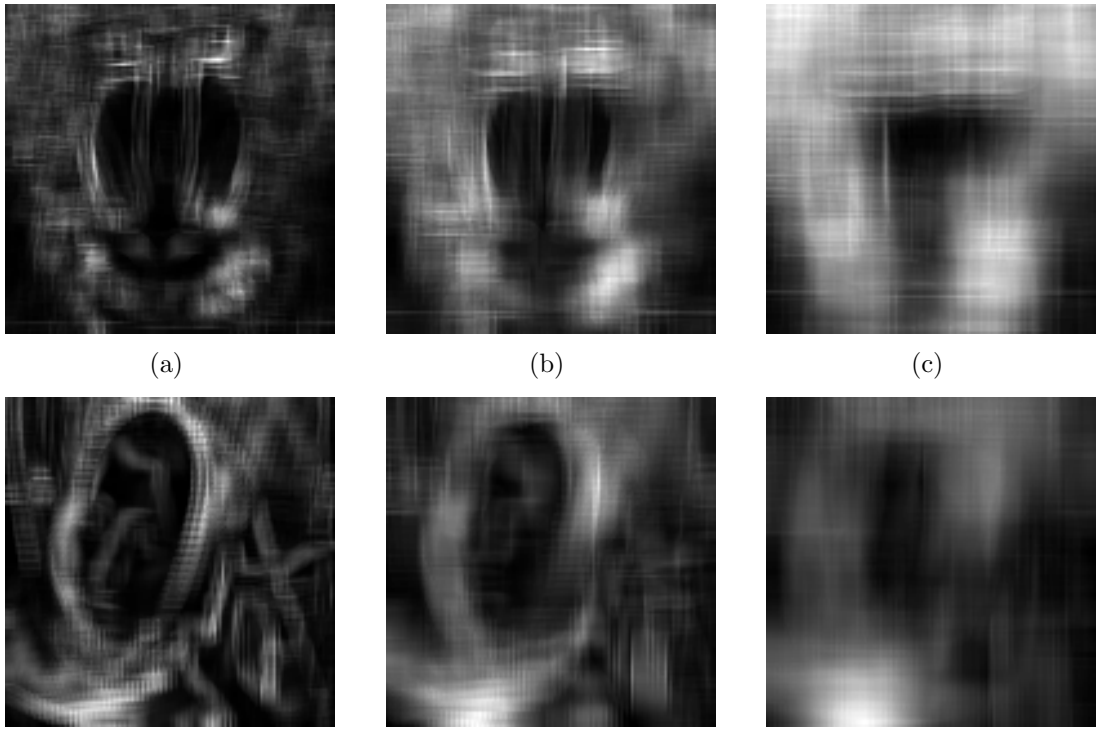
3. Build the steerable pyramid for I_0.

4. Perform a decomposition and obtain statistic parameters for each level.

Then, with the combination of the obtained parameters, we can obtain a function to decide if a pixel into I_0 belongs to a texture area or not. Simoncelli's method is a complex procedure, hence we let the reader to get inside by reading the former bibliographical references, specially on [84]. Results on the efficiency of Simoncelli's method can be shown in Figure 2.4. Empirically, it has been found that a window size of 16 pixels gives good results.

2.5 Quality metrics

Quality metrics would be described as all the tools that help to state a comparison between a target image (for instance, an image that have been distorted by any process) and a reference one. A quality metric must give a quantized measure based on an objective or subjective criteria (not strictly analytic). In order to be able to design reliable quality metrics, it is necessary to understand what "quality" means to the viewer. Viewers' enjoyment when looking at an image or video depends on many factors. One of the most important is of course the content and material. Provided the content itself is at least "watchable",



(a) (b) (c)

(d) (e) (f)

Figure 2.4: Simoncelli's texture detector performance. Parameters have been set as $N_{sc} = 2$, $N_{or} = 4$ and $N_a = 7$. For the images (a), (b) and (c), there have been calculated the texture over windows of 8,16 and 32 pixels on the image BABOON respectively. In the images (d), (e) and (f), the same procedure has been performed over BARBARA image.

visual quality plays a prominent role. Research has shown that perceived quality depends on viewing distance, display size, resolution, brightness, contrast, sharpness, colorfulness, naturalness and other factors ([1],[56],[80]).

It is also important to note that perceived quality is not necessary equivalent to *fidelity*, i.e. the accurate reproduction of the original. For example, sharp images with high contrast are usually more appealing to the average viewer [81]. Likewise, subjects prefer slightly more colorful and saturated images despite realizing that they look somewhat unnatural [77],[95].

Most "quality" metrics are actually fidelity metrics based on the comparison of the distorted image with a reference and neglect these phenomena. The reason for this is that without any reference it is very difficult for a metric to tell apart distortions from desired content, whereas humans usually are able to make this distinction from experience.

In the next sections, a short review on the pixel-based metrics is done as well as its unavailability to perform a quality measure according with the Human Visual System. Other metrics have been proposed [68] taking in consideration the peculiarities of the

Human Visual System and one of them, the MPQM-metric [6], will be introduced.

2.5.1 Pixel-Based Metrics

The mean squared error (MSE) and the peak signal-to-noise ratio (PSNR) are the most popular difference metrics in image and video processing. The MSE is the mean of the squared differences between the gray-level values of pixels in two pictures I and $\tilde{I}$:

$$\mathrm{MSE} = \frac{1}{N_x N_y} \sum_{x=0}^{N_x-1} \sum_{y=0}^{N_y-1} \left[I(x,y) - \tilde{I}(x,y) \right]^2, \tag{2.40}$$

for pictures of size $N_x \times N_y$. The average difference per pixel is thus given by the root mean squared error $\mathrm{RMSE} = \sqrt{\mathrm{MSE}}$.

The PSNR in decibels is defined as:

$$\mathrm{PSNR} = 10 \log \frac{m^2}{\mathrm{MSE}}, \tag{2.41}$$

where m is the maximum value that a pixel can take (e.g. 255 for 8-bit images). Note that MSE and PSNR are well-defined only for luminance information; there is no agreement on the computation of these measures for colour images.

Technically, MSE measures image difference, whereas PSNR measures image fidelity, i.e. how closely an image resembles a reference image, usually the uncorrupted original. The popularity of these two metrics is due to the fact that minimizing the MSE (or maximizing the PSNR) is equivalent to maximum likelihood estimation for independent measurement error with normal distribution. Besides, computing MSE and PSNR is very easy and fast. Because they are based on a pixel-by-pixel comparison of images, however, they only have a limited, approximate relationship with the distortion or quality perceived by human observers. In certain situations, the subjective image quality can be improved by adding noise and thereby reducing the PSNR. Dithering of color images with reduced color depth, which add noise to the image to remove the perceived banding caused by the color quantization, is a common example of this. Furthermore, the visibility of distortions depends to a great extent on the image content, a property known as masking. Distortions are often much more disturbing in relatively smooth areas of an image than in texture regions with a lot of activity, an effect not taken into account by pixel-based metrics. Therefore the perceived quality images with the same PSNR can actually be very different (see Figure 2.5). This phenomena is known as *masking*.

2.5.2 Masking

Masking occurs when a stimulus that is visible by itself cannot be detected due to the presence of another. Sometimes the opposite effect, facilitation, occurs: a stimulus that is not visible by itself can be detected due to the presence of another. Within the framework of image processing it is helpful to think of the distortion or coding noise being

masked (or facilitated) by the original image or sequence acting as background. Masking explains why similar distortions are disturbing in certain regions of an image while they are hardly noticeable elsewhere (Figure 2.5). Several different types of spatial masking can be distinguished [57],[92] but this distinction is not clear-cut.The terms *contrast masking*, *edge masking*, and *texture masking* are often used to describe masking due to strong local contrast, edges, and local activity, respectively.

2.5.3 MPQM Metric

Moving Pictures Quality Metric (MPQM) was introduced first in [6] as a quality measure that takes into account the human perception. This metric is based on a multi-channel model of human vision [15],[19] and a posterior masking of the input image. This multi-channel decomposition of the input image is done taking into account the responses of the individual visual receptive cells (V1). An analytic form of this norm is:

$$\text{MPQM} = \left(\frac{1}{N} \sum_{c=1}^{N} \left(\frac{1}{N_x N_y} \sum_{x=1}^{N_x} \sum_{y=1}^{N_y} |e[x,t,c]| \right)^{\beta} \right)^{\frac{1}{\beta}}, \tag{2.42}$$

where $e[x, y, c]$ is the masked error signal at position (x, y) and in the channel c; N_x and N_y are the horizontal and vertical dimensions of the image; N is the number of channels. The exponent of this Minkowski summation is β and has a value of 4, which is close to the human visual behavior [91].

The only drawback of this metric is its high correlation with the PSNR metric.

(a) (b)

Figure 2.5: Two images with identical PSNR or 33.2 dB. The same amount of noise has been added to a rectangular area at the top on the left and at the bottom on the right. The noise is much more visible in the sky than on the river due to strong masking, which PSNR does not take into account.

Chapter 3

Adaptive Greedy Approximations

3.1 Introduction

For data compression applications and fast numerical methods it is important to accurately approximate functions from a Hilbert space $\mathcal{H}$ using a small number of vectors from a given family $\{g_\gamma\}_{\gamma\in\mathbf{\Gamma}}$. The standard problem in this regard is the problem of M-term approximation where one fixes a basis and looks to approximate a target function f by a linear combination of M terms of the basis. For any $M > 0$, we want to minimize the aproximation error

$$\epsilon\left(M\right) = \|f - \tilde{f}\| = \left\|f - \sum_{\gamma\in I_M} c_\gamma g_\gamma\right\|, \tag{3.1}$$

where $I_M \in \mathbf{\Gamma}$ is the subspace formed by the M vectors that approximate our function f, c_γ are the ponderation coefficients and $\|\cdot\|$ is a general norm.

When the basis is orthogonal (a wavelet basis for instance), then, this type of approximation is the starting point for compression algorithms. In this special case, when an orthogonal basis $\{\psi_k\}_{k=1}^N \in \mathcal{H}$ ($\dim\mathcal{H} = N$) is taken to perform our M-term decomposition, this decomposition will be unique

$$f = \sum_{k=1}^{M} c_k\left(f\right)\psi_k + \epsilon\left(M\right) \qquad M < N, \tag{3.2}$$

where the coeficients $\{c_k\left(f\right)\}_{k=1}^M$ are the set of the Fourier coefficients of f, that is, the set of M vectors which have the largest inner products within $\langle f, \psi_k\rangle_{k=1}^N$. The problem of M-term approximations with regard to a basis has been studied thoroughly in [26],[24],[23].

One way to greatly improve these approximations consists in enlarging the collection $\{g_\gamma\}_{\gamma\in\mathbf{\Gamma}}$ beyond a basis. This enlarged, redundant family of vectors will be called dictionary. To be more precise, we define a dictionary as a family $\mathcal{D} = \{g_\gamma\}_{\gamma\in\mathbf{\Gamma}}$ of vectors in a $N-$dimensional Hilbert space $\mathcal{H}$, where the cardinality of $\mathcal{D}$ is P and $P > N$. All the vectors of $\mathcal{D}$ accomplish that $\|g_\gamma\| = 1$ and the finite linear expansions of $\mathcal{D}$ are dense in $\mathcal{H}$ ($\overline{span}\mathcal{D} = \mathcal{H}$) [63]. For our purposes, the application of adaptive greedy approximations

to image processing, we take $\mathcal{H} = \boldsymbol{L}^2(\mathbb{R}^2)$ [1] and we will call the vectors belonging to $\mathcal{D}$ as *atoms*.

Under an overcomplete basis (dictionary) the decomposition of a signal is not unique and this redundancy can offer some advantages (and also few drawbacks). One is that there is greater flexibility in capturing structure in the data. For example, if a signal is largely sinusoidal, it will have a compact representation in a Fourier basis. Similarly, a signal composed of chirps is naturally represented in a chirp basis. Combining both of these bases into a single overcomplete basis would allow compact representations for both types of signals [14],[62],[27]. It is also possible to obtain compact representations when the overcomplete basis contains a single class of basis functions, for instance: an overcomplete Fourier basis, with more than the minimum number of sinusoids, can compactly represent signals composed of small number of frequencies.

When the dictionary is redundant, finding a family of M vectors that approximates f with an error close to the minimum is clearly not achieved by selecting the vectors that have maximal inner products with f [21]. It is proven in [21] that for general dictionaries the problem of finding an M-element optimal approximations belongs to a class of computationally intractable problems: the set of NP-hard problems. That means that there is no known polynomial time algorithm that can compute the approximation $\tilde{f}$ that minimizes $\|f - \tilde{f}\|$ [18],[41].

Because of the difficult of computing optimal expansions, we turn to suboptimal algorithms: pursuit algorithms or adaptive greedy algorithms. These algorithms reduce the computational complexity by searching for efficient but non-optimal approximations. Within this family of algorithms we can enumerate Matching Pursuit (with its variants) and Basis Pursuit, among others [63],[27].

Under certain circumstances the approximation given by the Matching Pursuit algorithms can achieve *sparse* characteristics due to the fact that M (the number of terms to make this approximation) is much smaller than the dimension. A sufficient condition is to have at least two orthogonal vectors in the dictionary and, at certain point of our decomposition, be able to decompose the residual part of the approximation as a linear combination of those two orthogonal vectors to achieve zero error. The sparseness con-

[1] Notation:
The space $\boldsymbol{L}^2(\mathbb{R}^2)$ is the Hilbert space of complex valued functions such that

$$\|f\| = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} |f(x,y)|^2 \, dx \, dy$$

The inner product of $(f,g) \in \boldsymbol{L}^2(\mathbb{R}^2)$ is defined by

$$\langle f, g\rangle = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(x,y)\,\overline{g}(x,y)\, dx\, dy$$

And the norm is defined as

$$\|f\| = \langle f, f\rangle^{1/2}$$

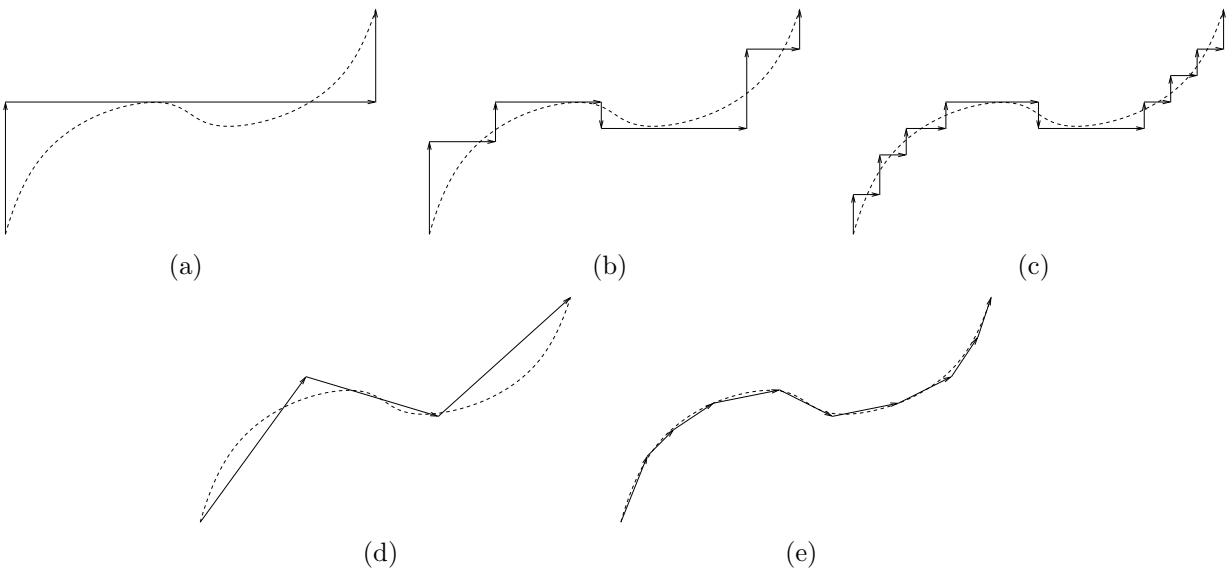


Figure 3.1: Example of representations. In the figures (a), (b) and (c) we have the decomposition over a orthogonal basis of a curve with 3, 9 and 17 terms respectively; in the figures (d) and (e) we have the decomposition of the same curve over an overcomplete dictionary with 3 and 9 terms. Using an overcomplete dictionary we capture the structure of the curve with much more less terms than with a decomposition using an orthogonal basis.

straint refers to the requirement that to represent the approximation function $\tilde{f}$ we must have as few c_k coefficients as possible [76]. Furthermore, it is proved that Matching Pursuit produces a (ϵ,M)-Sparse[2] approximation with exponential decay of the error [63],[88]. For any M-term approximation obtained with Matching Pursuit we have

$$\|f - \tilde{f}_{\mathrm{MP}}\| \leq \sqrt{1 + \frac{2\mu M^2}{(1 - 2\mu M)^2}} \|f - \tilde{f}_{\mathrm{Opt}}\|, \tag{3.3}$$

being μ the coherence of the dictionary $\mathcal{D}$ (see Section 4.2.1), $\|f - \tilde{f}_{\mathrm{MP}}\|$ the approximation obtained by Matching Pursuit and $\|f - \tilde{f}_{\mathrm{Opt}}\|$ the *optimal* approximation. That means that the error is bounded, hence the assumption of Matching Pursuit as a (ϵ,M)-Sparse problem is demonstrated.

[2] A (ϵ,M)-Sparse problem or a (ϵ,M)-Approximation refers to an approximation that achieves $\|f - \tilde{f}\| < \epsilon$ with M terms.

3.2 Matching Pursuit

3.2.1 Introduction

Matching Pursuit algorithm was introduced by Mallat and Zhang [62] giving examples for the application on unidimensional time-frequency signals (but it can be applied to any type of signal). This method produces a suboptimal function expansion by iteratively choosing the waveforms from a general dictionary (typically a rich collection of potential atoms in a Hilbert space) that best match the function's structures. The choice of the functions is performed through a progressive refinement of the signal approximation with an iterative procedure [21]. This method is closely related to the algorithms used in statistics [36].

The Matching Pursuit algorithms have already found applications in medicine [35] and image [4] and video coding [3] (though in video coding it is usually used to code the motion estimation errors). Other flavors of Matching Pursuit can also be found in [62] and [63] like the Orthogonalised Matching Pursuit that is able to achieve a zero estimation error by orthogonalizing the directions of projection, with a Gram-Schmidt procedure proposed by [73]. The resulting orthogonal pursuit converges with a finite number of iterations, which is not the case for a non-orthogonal pursuit. The price to be paid is the important computational cost of the Gram-Schmidt orthogonalization, though this is not used due to practical reasons (fast algorithms to perform this Orthogonal Matching Pursuit have been already proposed in [42]).

With this thesis we want to show that Matching Pursuit is much more efficient to do an image approximation than the usual methods used nowadays in the standard formats (DCT for JPEG [51] and wavelets for JPEG2000 [52]), so it is possible to transmit an image at lower bit-rate. Matching Pursuit, though results strongly depended on the choice of the dictionary(ies) used. In many applications, Gabor functions or symmetric dictionaries are used; we can greatly improve the results by using two or more dictionaries that catch more efficiently different features of the image (like edges or textures). Furthermore, in this thesis few Matching Pursuit hybrid schemes to perform decompositions over more than one dictionary taking into account the characteristics of the Human Visual System modelled by probability masks according to the limitations and peculiarities of our vision system will be exposed. This representations will not achieve best PSNR results than the standard Matching Pursuit algorithm described here but better results according with the visual quality.

3.2.2 Formulation

Matching Pursuit is a greedy algorithm that decomposes any signal belonging to a Hilbert space $\mathcal{H}$ into a linear expansion of waveforms that are selected from a redundant dictionary (or set of dictionaries) $\mathcal{D}$ of functions. These waveforms are iteratively chosen to best match the signal structures, producing a sub-optimal expansion. Vectors are selected one by one from the dictionary, while optimizing the signal approximation at each step k (this is the minimization $\|f - \tilde{f}\|_k$ with reference to $\|f - \tilde{f}\|_{k-1}$).

Let $\mathcal{D} = \{g_\gamma\}_{\gamma \in \mathbf{\Gamma}}$ be a dictionary of $P > N \times M$ vectors, with the properties cited above. This dictionary includes $N \times M$ linearly independent vectors that define a basis of the space $\mathbb{R}^{N \times M}$ of signals with size $N \times M$. The Matching Pursuit algorithm begins by projecting the target function f on a vector $g_{\gamma_0} \in \mathcal{D}$ and computing the residue Rf (see [62] and [63]):

$$f = \langle f, g_{\gamma_0} \rangle g_{\gamma_0} + Rf, \tag{3.4}$$

where Rf is the residual vector after approximating f in the direction of g_{γ_0}. Since we impose Rf to be orthogonal to g_{γ_0}:

$$\|f\|^2 = |\langle f, g_{\gamma_0} \rangle|^2 + \|Rf\|^2. \tag{3.5}$$

As we want to minimize $\|Rf\|^2 = \|f\|^2 - |\langle f, g_{\gamma_0} \rangle|^2$ we must choose $g_{\gamma_0} \in \mathcal{D}$ such that $|\langle f, g_{\gamma_0} \rangle|$ is maximum. In some cases, it is not computationally efficient to find the solution given by the Matching Pursuit algorithms, and a Matching Pursuit-suboptimal solution is computed instead:

$$|\langle f, g_{\gamma_0} \rangle| \geq \alpha \sup_{\gamma \in \Gamma} |\langle f, g_\gamma \rangle|, \tag{3.6}$$

where $\alpha \in (0, 1]$ is an optimality factor ($\alpha = 1$ means that we choose the optimal solution given by the Matching Pursuit method).

Into the next step, Matching Pursuit subdecomposes iteratively the residue Rf by projecting it on a vector of $\mathcal{D}$ that matches Rf at best. If we consider $R^0 f = f$ and we suppose the n-th order residue $R^n f (n \geq 0)$ has been computed, the next iteration will choose $g_{\gamma_n} \in \mathcal{D}$ such that:

$$|\langle R^n f, g_{\gamma_n} \rangle| \geq \alpha \sup_{\gamma \in \Gamma} |\langle R^n f, g_\gamma \rangle|. \tag{3.7}$$

With this choice $R^n f$ is projected onto g_{γ_n} and decomposed as follows:

$$R^n f = \langle R^n f, g_{\gamma_n} \rangle g_{\gamma_n} + R^{n+1} f, \tag{3.8}$$

where $R^{n+1} f$ and g_{γ_n} are orthogonal, so the quadratic module of the previous equation is:

$$\|R^n f\|^2 = |\langle R^n f, g_{\gamma_n} \rangle|^2 + \|R^{n+1} f\|^2. \tag{3.9}$$

From 3.8, we can see that the decomposition of f is given by:

$$f = \sum_{n=0}^{N-1} \langle R^n f, g_{\gamma_n} \rangle g_{\gamma_n} + R^N f, \tag{3.10}$$

and with the same principle we can also deduce from 3.9 that the module of the signal f is:

$$\|f\|^2 = \sum_{n=0}^{N-1} |\langle R^n f, g_{\gamma_n} \rangle|^2 + \|R^N f\|^2, \tag{3.11}$$

where $\|R^N f\|$ converges exponentially to 0 when n tends to infinity[3]:

$$\lim_{n\to\infty} \|R^n f\| = 0. \tag{3.12}$$

Hence

$$f = \sum_{k=0}^{\infty} \langle R^k f, g_{\gamma_k} \rangle g_{\gamma_k}, \tag{3.13}$$

and

$$\|f\| = \sum_{k=0}^{\infty} \left| \langle R^k f, g_{\gamma_k} \rangle \right|^2. \tag{3.14}$$

Only with Orthogonalised Matching Pursuit[4] [29], [63], [21] the residue is reduced to 0 in a finite number of iterations but, in most signal processing applications, the fact of having a non-zero residual is not relevant, due to the fact that the image distortion is under the visible threshold. Properties of Matching Pursuit can be fount in Appendix C.

3.2.3 Extensions

It is obvious that by choosing the atom that gives the largest absolute scalar product leads to maximize the PSNR ratio of the reconstructed image at each iteration. Few extensions could be suggested:

- Use of a different norm. Instead of maximizing the absolute value of the scalar product, other metrics could be used. A proposal would be to maximize a metric based on the human perception of images, that is, choose the atoms that are more suitable for our HVS at each iteration (MPQM,...).

- Choose more than one atom per iteration. Not all the atoms are localized in the same space area, hence, at the n-th iteration, all the atoms that not collide in space [60] could be chosen. This procedure will not lead to the same results given by the standard Matching Pursuit, but the compromise between speed and accuracy arises. With this method Matching Pursuit is speed-up and the final result do not differ in a noticeable way from the standard Matching Pursuit.

[3] See Appendix B for a detailed demonstration

[4] It is also true that certain types of signals (for example, signals composed by a linear combination of atoms) can achieve zero error without this method.

3.3 Full search Matching Pursuit via FFT

Matching Pursuit is a greedy algorithm that decomposes a signal over a redundant set of functions, the dictionary. As we have seen in the former section, the algorithm computes for each iteration all the scalar products $\langle R^k f, g_{\gamma_k} \rangle$ (in the k-th iteration) and then chooses the one with the largest absolute value. If we analyze in detail the computational load this calculations generate, for example for the dictionary $\mathcal{D}_{AR}$ formed by Anisotropic Refinement atoms (see Section 4.3 for more references about this dictionary) we will find that it is huge. For a square image of size $N_x \times N_x$, the number of scalar products between images to be done in the computation of one decomposition term is

$$N_x^2 \times \frac{1}{2} \left(3 \times \left(\log_2 \left(N_x\right) - 2\right) + 1\right)^2 \times 18 \sim O\left(N_x^2 \times \log_2^2 \left(N_x\right)\right), \tag{3.15}$$

equivalent to have this amount of operations per coefficient

$$O\left(N_x^4 \times \log_2^2 \left(N_x\right)\right). \tag{3.16}$$

Then, optimizations of this algorithms are required in order to reduce the computational load. But here arises another problem: when performing Matching Pursuit, it can be chosen between generating the atoms at each iteration to compute the scalar products or storing them in memory in order to save computational time. The option to generate and store the atoms in the memory is the optimal one in terms of speed but on the other hand there is the problem of memory capacity. Hence, there is a compromise between memory and speed. Moreover, to perform the Matching Pursuit decomposition without any optimization implies to store in memory all the functions of the dictionary (a prohibitive amount of memory!, for further details on the size of the dictionaries see Chapter 4). In the pursue of optimizations, suboptimal approximations of Matching Pursuit have been proposed by using genetic algorithms ([31], [32]) that reduce the computational load but the drawback of this type of techniques are the non repeatibility of the process, that is: if you apply twice the algorithm over the same image you obtain different results.

An optimization, proposed by [34], is to use the properties of the Discrete Fourier Transform to reduce the computational load (using also the Fast Fourier Transform) and reduce though the memory required to store the dictionary. This optimization is based on the property of the duality product-convolution of the DFT already introduced in the first chapter.

Here we do a detailed explanation. Let $\mathcal{D}$ be a dictionary defined by a set of parameters $\gamma = (\aleph, \boldsymbol{p})$ where $\aleph$ are the set of parameters concerning the shape of the atom (orientation,scaling,...) and $\boldsymbol{p} = (p_x, p_y)$ the point into the image where the atom will be centered. Let us define $\mathcal{V} \in \mathcal{D}$ the sub-dictionary generated by $\gamma = (\aleph, \underline{\mathbf{0}})$, the set of atoms centered in the middle of the image [5]. Strictly applying the Matching Pursuit algorithm, to find the most powerful atom at the n-th iteration of the process we should compute all the scalar

[5] For this expanation, let us consider an image bounded into $[-\frac{N_x}{2}, \frac{N_x}{2}] \times [-\frac{N_y}{2}, \frac{N_y}{2}]$.

products $|\langle R^n f, g_{\gamma_n}\rangle|$ with $g_{\gamma_n} \in \mathcal{D}$ and then choose the largest one. The search of the most powerful can be rewritten as the search of $z_{\gamma_n}(x,y) \in \mathcal{V}$ that maximizes

$$\max_{\gamma_n} |\langle R^n f, z_{\gamma_n}(x-p_x, y-p_y)\rangle|, \tag{3.17}$$

with $p_x \in [-\frac{N_x}{2}, \frac{N_x}{2}]$ and $p_y \in [-\frac{N_y}{2}, \frac{N_y}{2}]$. With a simple manipulation, the Equation 3.17 can be formulated in terms of a convolution operation

$$\max_{\gamma_n} \left\| R^n f * z_{\gamma_n}(x,y) \right\|, \tag{3.18}$$

bounded into the frame $[-\frac{N_x}{2}, \frac{N_x}{2}] \times [-\frac{N_y}{2}, \frac{N_y}{2}]$. At this point, by applying the duality product-convolution of the DFT it leads to

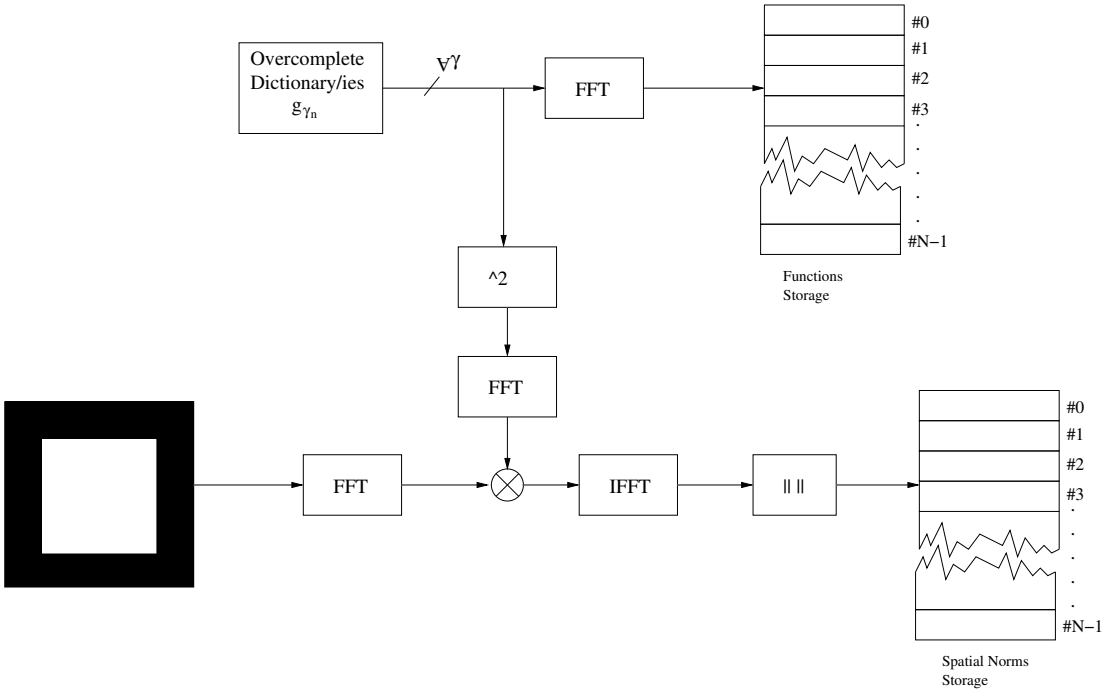
$$R^n f * z_{\gamma_n}(x,y) \xrightarrow{\mathcal{F}} \widehat{R^n f} \cdot \hat{Z}_{\gamma_n}(x,y), \tag{3.19}$$

where $\widehat{R^n f}$ and $\hat{Z}_{\gamma_n}(x,y)$ are the Fourier transforms of $R^n f$ and $z_{\gamma_n}(x,y)$ respectively. Finally, to search the most powerful atom by using this DFT based method can be written as

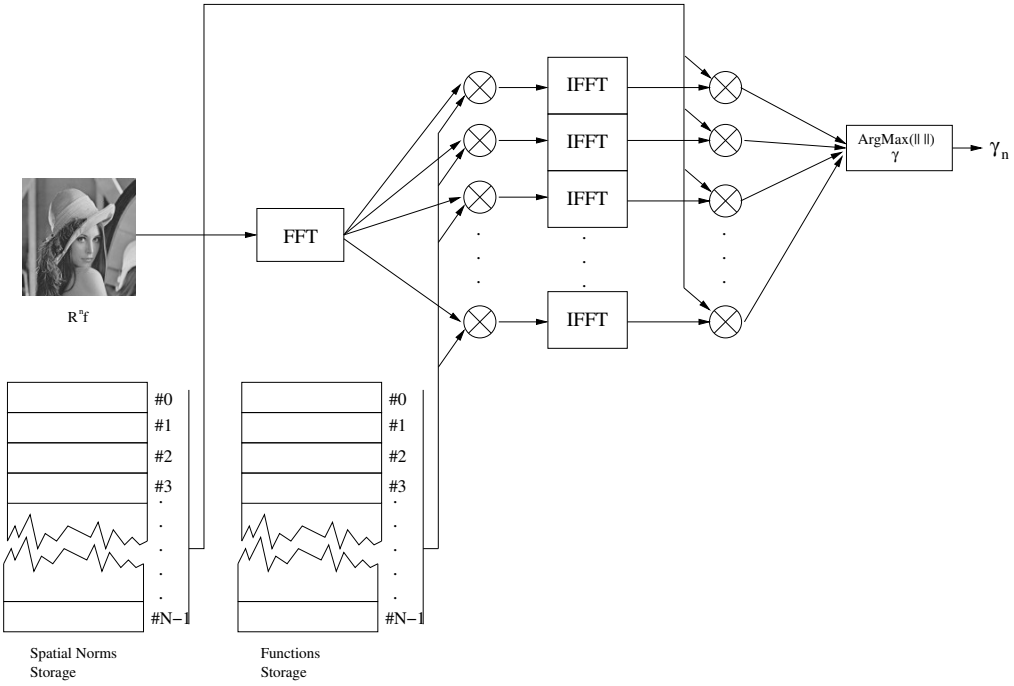
$$\max_{\gamma_n} \left\| R^n f * z_{\gamma_n}(x,y) \right\| = \max_{\gamma_n} \left\| \mathcal{F}^{-1} \left\{ \widehat{R^n f} \cdot \hat{Z}_{\gamma_n}(x,y) \right\} \right\|. \tag{3.20}$$

This full-search method proposed below takes advantage of the FFT usage, saving a lot of computational load. Exactly, the Matching Pursuit complexity for one atom is reduced to

$$O\left(N_x^2 \times \log_2^3(N_x)\right). \tag{3.21}$$



(a)



(b)

Figure 3.2: FFT implementation of Matching Pursuit algorithm. In the figure (a) is depicted the initialization process: the frequency spectrum of the atoms belonging to the dictionary are stored in memory as well as a ponderation mask to avoid the effects of those atoms placed in the borders of the image. Those atoms do not have unitary norm, hence its scalar product must be compensated to maintain the properties of the dictionary. In the figure (b), the FFT-Matching Pursuit implementation is shown (see Equation 3.20).

3.4 Weak Matching Pursuit

As we will see in Chapter 5, this thesis is focused on the use of probability masks to aid Matching Pursuit standard algorithm to deal efficiently with more than one dictionary. Furthermore, a simple model of the perception by the Human Visual System of these results will be analyzed and, finally, an Hybrid Matching Pursuit coder is presented. This coder would achieve both the sparsity and decomposition of the Matching Pursuit algorithm and a representation of the image that would be more adapted to our early visual system (even having a worst PSNR ratio in comparation with the pure Matching Pursuit algorithm).

The main idea of this model is to ponder the scalar products obtained by the standard Matching Pursuit algorithm by a probability mask (or even more masks in the case of a multi-dictionary decomposition) and then choose the pondered atom with bigger scalar product (note than once the pondered atom is chosen, the reconstruction and updating of the residual part $R^n f$ is performed with the original scalar product unless we would lead to a senseless result). Hence, it must be justified that, even with this masking of the coefficients, the algorithm leads to stable solutions, it achieves a perfect signal representation when an infinite number of atoms is used. This modification of the Matching Pursuit was introduced by [86] and is known as Weak Matching Pursuit.

3.4.1 Formulation

Let $\tau = \{t_k\}_{k=1}^{\infty}$, $0 \geq t_k \geq 1$ be a weakness sequence. We can define the Weak Matching Pursuit (or Weak Greedy Algorithm as noted in [86]) similarly as the Matching Pursuit algorithm:

1. Define $R^0 f^\tau = f$.
2. Take $g^\tau_{\gamma_n} \in \mathcal{D}$ satisfying:
$$\left|\langle R^n f^\tau, g^\tau_{\gamma_n}\rangle\right| \geq t_n \sup_{\gamma\in\Gamma} \left|\langle R^n f^\tau, g^\tau_{\gamma}\rangle\right|. \tag{3.22}$$
3. As we set in 3.8, we actualize the residue by projecting it onto $g^\tau_{\gamma_n}$:
$$R^n f^\tau = \langle R^n f^\tau, g^\tau_{\gamma_n}\rangle g^\tau_{\gamma_n} + R^{n+1} f^\tau. \tag{3.23}$$
4. Then, we can construct our approximation
$$f = \sum_{n=0}^{N-1} \langle R^n f^\tau, g^\tau_{\gamma_n}\rangle g^\tau_{\gamma_n} + R^N f^\tau. \tag{3.24}$$

Once Weak Matching Pursuit has been formulated, we must analize under which conditions the sequence $\tau = \{t_k\}_{k=1}^{\infty}$ leads to a stable solution of the decomposition. A first sufficient condition on τ for the convergence of any f in our Hilbert space $\mathcal{H}$ is
$$\liminf_{k\to\infty} \tau = 0. \tag{3.25}$$

Also, in the class of monotone sequences $\tau = \{t_k\}_{k=1}^{\infty}$, $1 \geq t_k \geq 0$, the condition imposed by Eq.3.25 is accomplished and, hence, the convergence proved. This particular type of sequences are those that will arise in further explanations related with the use of probability masks in the Matching Pursuit process. A full non-trivial demonstration of the conditions imposed over τ to lead to a stable solution can be found in [87].

3.5 Computing Implementation: Parallel MPI/LAM and Sparse Routines

To compute the Matching Pursuit decomposition, a huge computational effort has to be performed and a large amount of memory is required as well. In fact, to perform this decomposition over a standard machine (let us say a Pentium III at 1.5 Ghz[6]), the time required is prohibitive (even with an optimized code it takes few days to obtain a result). Moreover, regarding with the matter of the memory, even using the Matching Pursuit-Full Search into the Fourier domain that achieves a lower memory demand, it is still intractable. Then, both optimizations for memory and computational load are strongly required to perform a decomposition within an acceptable time and with a machine without an overdimensioned memory system.

If we look carefully to the properties of the atoms that will be used to perform the Matching Pursuit decomposition (see Chapter 4), we notice that all them are well localized both in space and frequency. Hence, as the Fourier Transform of the atoms is stored into the memory (according with the procedure described in the Section 3.3), it would be possible to take advantage of its sparsity (well localization in frequency). In fact, if the functions are stored by using a run-length compression this purpose is achieved. The routines proposed by [25] are the solution to this problem.

Regarding with the problem of the computation load, strategies to reduce this time must be found and parallel computing brings us efficient tools to carry out this problem. A good solution is the Message Passing Interface (MPI) ([67],[50]) that allows to write a distributed code that is able to perform a Matching Pursuit decomposition in a fraction of the time needed by a single machine (actually, $T_{MPI} = \alpha \frac{1}{k} T_{SINGLE}$ being k the number of slave machines and α a scaling factor). Even though, this library incorpores a complete set of functions to control the synchronization and share of data between all the machines very suitable for our purposes. Furthermore, once again, it is possible to take advantage of the sparsity presented by the atoms used here. The routines presented in [25] also allow to perform efficient products between sparse images, saving around a 10% of the initial computational load.

Finally, the implementation of the Matching Pursuit parallelization is depicted in Figure 3.5. The idea that underlies in this parallelization is to split the computational load in a set of k slave-machines that send their partial results to master-machine that combines them. A short review on the function of the whole distributed system is:

[6]Note for the reader in the future: This data is taken from Summer 2003.

- **Balanced load**.The computational load will be distributed in the sense that every slave-machine will perform a full-search Matching Pursuit over a portion of the original dictionary. Hence, the first step is to split the original dictionary in k portions and assign each one to the slave-machines. But this process must be done smartly, that is, to distribute the functions in order to balance the computational load over them. As sparse routines [25] are used, it will not take the same time to perform a product with a small atom than with a big atom. Hence, the balance of the load must be done in the sense to assign approximately the same number of atoms of the same type to each slave. In this way, the time taken by each slave will be approximately the same then there will not be a bottle-neck that would slow down the system.

- **Partial Matching Pursuit**. The k slaves have its own copy of the residual image over which the Matching Pursuit decomposition is done[7]. Then, each slave performs a Full Search Matching Pursuit over the portion of the dictionary assigned to it and chooses the atom that gives the largest absolute scalar product. Afterwards, these values are sent to the master-machine and it chooses among them the best one. Then, the chosen atom is stored to create the reconstruction and is also sent back to all the slaves to update their residual images for the next iteration.

- **Synchronization**. This issue is always extremely important when dealing with multiple machine interfaces. This process is carried out by the master-machine that sets the start time for all the slaves. Hence, this is a synchronous system, otherwise, an asynchronous system would lead to dead-lock situations (that is when the system enters in a situation where every machine is waiting for another and it collapses).

[7]When dealing with MPI systems the communications between machines must be minimum in order to minimize the effects of the communication channel and improve the performance. Hence, the optimal way to operate is to make each slave to store its own copy of the residual image instead of transmitting it.

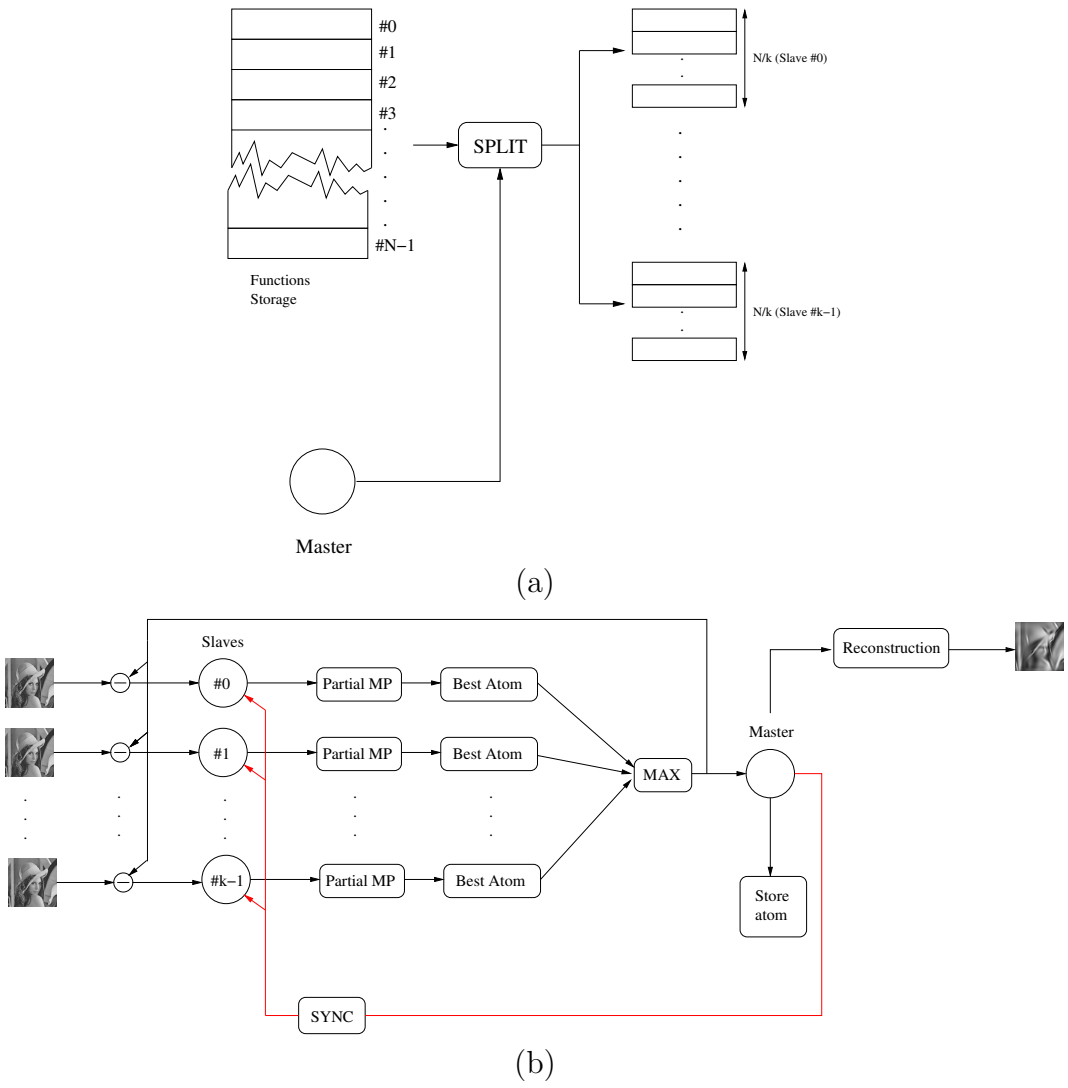


Figure 3.3: MPI/LAM parallel implementation of Matching Pursuit. In subfigure (a), the splitting process of the main dictionary into sub-dictionaries for each slave-machine. The master-machine takes care to make a uniform division of the main dictionary in order to balance the load of Matching Pursuit (for example, in the number of functions and the type of them for each slave-machine). Subfigure (b) depicts the parallelized process of Matching Pursuit.

Chapter 4

Analysis of the dictionaries

4.1 Dictionary usage

Matching Pursuit greedy approximation of functions depends exclusively on the election on the dictionary/ies the decomposition will be done over. Hence, a careful study over the design and performance of the dictionary/ies must be done. Many parameters are involved in the design of a dictionary: shape of the atoms, orientation, scales, frequency (i.e. the case of Gabor atoms),... The values those parameters take will define the size, properties and performance of the dictionary. For instance, a careful design of the dictionary would lead us to reduce its size by eliminating atoms that would never be used (for example, analyzing the histogram of the atoms, see Section **??**). Furthermore, the redundancy of the dictionary can be controlled (this is the case of the Gabor dictionary presented below).

In this chapter, performance methods like the coherence and the Babel function are presented, as well as the description and design criteria for the three dictionaries that have been used for our experiments: the Anisotropic Refinement Dictionary $\mathcal{D}_{AR}$, the Gaussian Dictionary $\mathcal{D}_G$ and the Gabor Dictionary $\mathcal{D}_{Ga}$.

4.2 Dictionary performance

4.2.1 Coherence

Once the dictionaries are already chosen, a quality parameter might be defined to evaluate how good our choice is. The most fundamental quality parameter associated with a dictionary is the *coherence* μ [88], defined as

$$\mu = \max_{j \neq k} \left| \langle g_{\gamma_j}, g_{\gamma_k} \rangle \right| \qquad g_{\gamma_k}, g_{\gamma_j} \in \mathcal{D}. \tag{4.1}$$

Roughly speaking, this number measures how much two vectors in the dictionary look alike. This parameter is not a definitive way to evaluate the performance of the dictionary since it only reflects the most extreme correlations in the dictionary. Nevertheless, it is

easy to calculate, and it captures well the behavior of uniform dictionaries. In the next section a new quality parameter that avoids this problem is presented: *the Babel function* [88].

Let be $\mathcal{H}$ a Hilbert space of dimension N and $\mathcal{D}$ an arbitrary dictionary formed by P atoms. From the coherence function definition it is obvious that an orthogonal basis has coherence $\mu = 0$ and it is proved, for general dictionaries, that a lower bound on the coherence is [88]

$$\mu \geq \sqrt{\frac{P-N}{N\,(P-1)}}. \tag{4.2}$$

If this parameters is small enough, we can call a dictionary to be *incoherent*. A table of the μ's parameters for the dictionaries used in this thesis can be computed:

	μ
Gaussian Dictionary $\mathcal{D}_G$	0.9836
Gabor Dictionary $\mathcal{D}_{Ga}$	0.9400
AR Dictionary $\mathcal{D}_{AR}$	0.9964

and we can appreciate that our dictionaries show a high redundancy, so they are not incoherent. Also, furhter works have been developed on the properties of this parameter [62],[28] and its relations with different types of dictionaries [47],[44],[43].

4.2.2 The Babel function

The Babel funcion introduced in [88] is a suitable paramter to make a deeper analysis of the structure of a dictionary. While the coherence parameter does not offer a very subtle description of a dictionary since it only reflects the most extreme correlations between atoms, a new quality measure must be defined. When most of the inner products are tiny, the coherence can be downright misleading (a wavelet packed dictionary exhibits this type of behavior). The Babel function measures the maximum total coherence between a fixed atom and a collection of other atoms. In a sense, the Babel function indicates how much the atoms are "speaking the same language". It is much simpler to distinguish Catalan from Russian than it is to distinguish Catalan from Spanish.

The Babel function is defined as

$$\mu_1(m) = \max_{|\Lambda|=m} \max_{\psi} \sum_{\Lambda} |\langle \psi, g_\lambda \rangle|\,, \tag{4.3}$$

where the vector ψ ranges over the atoms indexed by the complementary of Λ. In other words, let be g_{λ_j} the set of vectors belonging to the set Λ and g_{λ_k} those that not belong, then the Babel function can be rewritten as

$$\mu_1(m) = \max_{|\Lambda|=m} \max_{k \notin \Lambda} \sum_{j \in \Lambda} \left|\langle g_{\lambda_k}, g_{\lambda_j} \rangle\right|. \tag{4.4}$$

A close examination of the formula shows that $\mu_1(1) = \mu$ and that μ_1 is an increasing function. Nevertheless, the computation of this formula implies an extremely heavy computational load for a general dictionary. Even with a dictionary that has an analytical form, the computation is intractable (in few cases is possible to find the analytical form of the Babel function [88]). For our three dictionaries, the only one that allows a feasible approximation of the calculation of $\mu_1(m)$ is the Gaussian Dictionary $\mathcal{D}_G$. In fact, we calculate a lower bound of this function (see Appendix B) for $\mathcal{D}_G$ depicted in the Figure 4.1. This functions gives an idea about the redundancy of the dictionary. In future work, the knowledge of this function (or a lower-upper bound) could be taken into account to design more efficient dictionaries.

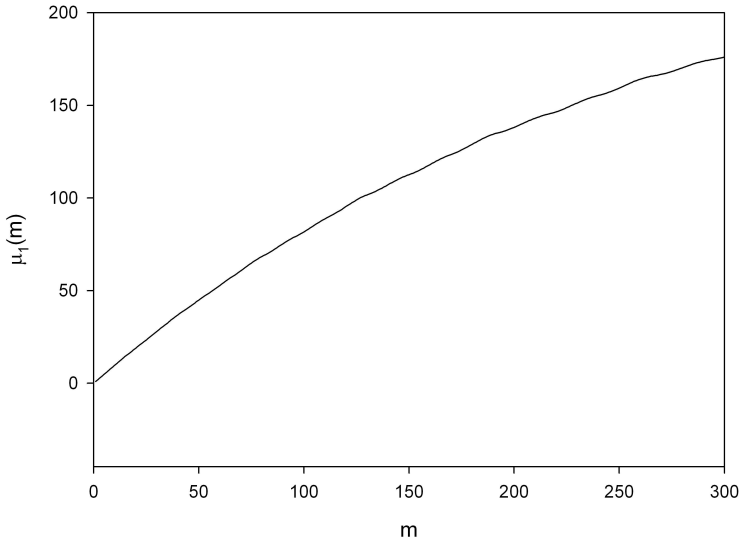


Figure 4.1: Babel function $\mu_1(m)$ for a the Gaussian dictionary $\mathcal{D}_G$.

4.3 Designing a Dictionary Set

In a Matching Pursuit process the decisive element is the election of the dictionary (or set of dictionaries) that will be used to represent our signal. Extensive literature on the design of dictionaries based on several types of atoms exists: Gabor atoms [62], Anisotropic Refinement [89],[31], chessboard atoms [60], Walsh atoms [90],... or even dictionaries consisting in the union of few orthonormal bases [44]. In the following sections, the dictionaries used for our simulations are presented, as well as a design analysis.

The first implementation of Matching Pursuit in [31] was based on the substraction of the image baseband (via a downsampling process) and, then, coding the result (an image basically formed by the high frequencies so, edges and textures) using the MP greedy approximation. This scheme is rather simple: it can be done better in the sense that the

method does not pay attention to the textures of the image. An improved scheme might be proposed based on the use of several dictionaries to code the diverse features of the the image. So let us define a dictionary framework:

- A dictionary to code the *low frequencies* that is: the *baseband* of the image. In previous schemes [31], this baseband was coded directly by coding the downsampled image (via a determined factor). In the scheme proposed here, the baseband will be coded by a dictionary based on Gaussian functions that catches pretty efficiently the flat areas of the image (similarly to what has been done in [34].
- A dictionary to code the *edges* and *contours*. This dictionary will be based on Anisotropic Refined Gabor atoms (AR) as done previously in [31],[89].
- A dictionary to code the *textures*. It is proved that to code a texture (a pattern) directly using the AR atoms is extremely inefficient due to the fact that the Matching Pursuit algorithm wastes a lot of atoms to code this pattern (see Figure 4.3). This new dictionary will be based on Gabor Atoms (so, Gaussians modulated by various frequencies according to few constrains we will impose).

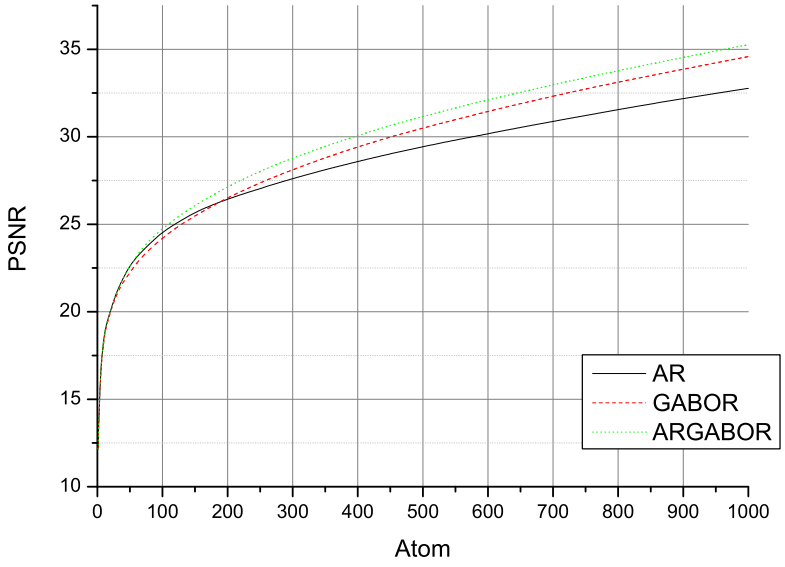
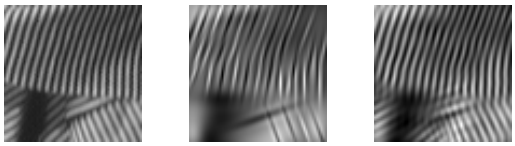
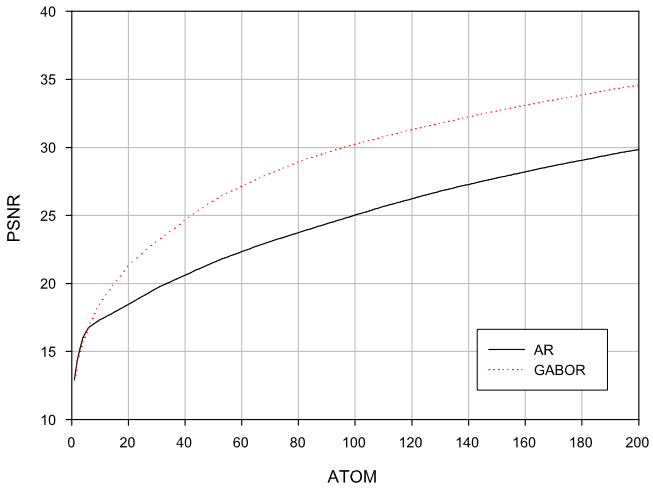


Figure 4.2: PSNR ratio for Matching Pursuit approximations using various dictionaries types over BARBARA image. It is shown that the combined use of Anisotropic Refinement and Gabor dictionaries (ARGABOR) leads to better PSNR ratios than the use of single dictionaries (AR or GABOR) because it catches better the structures of the image.



(a) (b) (c)



(d)

Figure 4.3: Example of texture coding with different dictionaries. Subfigure (a) corresponds to a detail of BARBARA and (b) and (c) to the Matching Pursuit coding with 50 AR atoms and GABOR atoms respectively. As reflected in the PSNR diagram (d) the results show the suitability of GABOR atoms to code textures.

We will join these three dictionaries to make a macro-dictionary. The first approach to the use of these dictionaries will be to perform the Matching Pursuit decomposition computing all the scalar products between them and the target image. Nevertheless, other Matching Pursuit schemes present in this work will apply these dictionaries separately in localized areas by previously detecting whether a feature is present or not in a determinated region of our image.

As our dictionary increases in number of functions (in the sense we join our three dictionaries), the PSNR ratio improves due to the fact that the algorithm have more atoms to catch the structures of the image. In fact, when PSNR is taken as our quality measure (objective), it increases by using the three dictionaries. But, in the scheme of Matching Pursuit presented in this thesis, we will take into account Human Visual System features so, the PSNR ratio will not be our reference quality measure. Instead of it, subjective measures and visual quality measures will be used. However, the results of the PSNR curve for the three dictionaries are depicted in Figure 4.2 showing its performance.

4.3.1 Dictionary $\mathcal{D}_{AR}$ based on AR atoms

The use of dictionaries based on Anisotropic Refinement of Gabor atoms was introduced by [89] to code the edges of the images more efficiently. Furthermore, it has been shown in [71] and [33] that the use of Anisotropic Refinement atoms are more suitable for edge image coding due to the fact that the Human Visual System (HVS) is adapted to this type of functions [72]. In this way, this type of atoms seems to be the right choice to code the edges of images.

An anisotropic refined Gabor atom is defined as a combination of a gaussian in an axis and its second derivative in the other axis

$$g_\gamma(x,y) = K\left(4x^2 - 2\right)e^{-(x^2+y^2)}, \tag{4.5}$$

where K is a normalization constant to have unitary norm. Assuming that our image has a size of $N_x \times N_y$ pixels, we can define the set of parameters $\gamma = (\boldsymbol{p}, \boldsymbol{s}, \theta)$ necessary to generate the whole dictionary:

$\boldsymbol{p} = [p_x, p_y]$ where $p_x \in [0, N_x), p_y \in [0, N_y)$	translations
$\boldsymbol{s} = [s_x, s_y]$	scaling factors
$\theta \in [0, \pi)$	rotation,

with $\boldsymbol{p}$ the translation vector that will set the center of the atom into the image, $\boldsymbol{s}$ where s_x is the dilation in the x axis and s_y is the dilation in the y axis and θ the rotation angle.

At this point, we have to choose our parameters carefully to be sure that this set of atoms have an associated family that is a frame of $\boldsymbol{L}^2(\mathbb{R}^2)$ [63]. If $\Delta\theta$ and Δs are small enough finite linear expansions of space-frequency atoms are dense in $\boldsymbol{L}^2(\mathbb{R}^2)$, hence this dictionary is also complete and, then, valid for image coding. To satisfy these conditions we have chosen our parameters in this way:

- $\Delta\theta = 10^0$.
- $\{s_x, s_y\} \in [0, NN \cdot (\lfloor \log_2(N) \rfloor - 3)] \in \mathbb{Z}$ where $NN \in [1, \log_2(N)] \in \mathbb{Z}$, $N = \min(N_x, N_y)$ and $s_x > s_y$.

Once we have defined the parameters to generate the atom's family, we can define the procedure to create the atom. This procedure is non-commutative, so, the order is fixed and its application is:

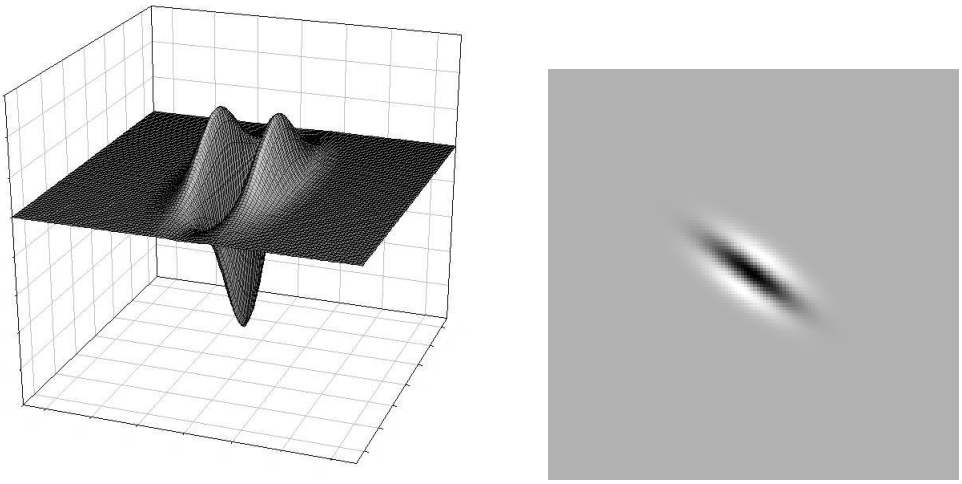
1. Apply the translation by $[p_x, p_y] \in \mathbb{Z}^2$.
2. Rotate by θ the translated atom.
3. Scale the translated and rotated atom by $\sigma_x = 2^{\frac{s_x}{NN}}$ and $\sigma_y = 2^{\frac{s_y}{NN}}$ in the axis x and y respectively.

This leads to compute:

$$x_\gamma = \frac{(x - p_x)\cos(\theta) + (y - p_y)\sin(\theta))}{2^{\frac{s_x}{NN}}} \tag{4.6}$$

$$y_\gamma = \frac{(x - p_x)\sin(\theta) - (y - p_y)\cos(\theta))}{2^{\frac{s_y}{NN}}}, \tag{4.7}$$

and $g_\gamma = g(x_\gamma, y_\gamma)$.



(a) (b)

Figure 4.4: Anisotropic refined Gabor atom.

4.3.2 Our choice

For our experiments, we took into account the experiences related in [31] and our own empirical results. The obtained parameters that will be used in our experiments are:

Translations The translations must cover the whole image, hence:

$$[0, 0] \leq [p_x, p_y] \leq [N_x - 1, N_y - 1]. \tag{4.8}$$

NN Empirically [31], it has been shown that the quality of the coded image will increase with NN, but the size of the dictionary too. If NN is too small some artifacts (lines) will appear in the coded image, and if NN is too large, the computational load to perform Matching Pursuit becomes prohibitive. A good choice, based in our empirical experiences, is to take

$$NN = 3. \tag{4.9}$$

Scaling Factors The scaling factors are chosen in a way that the smaller scaling applied is one and the bigger is the size of the image. So:

$$2^{\frac{s_{i_{\min}}}{NN}} = 1 \quad \rightsquigarrow \quad s_i \geq 0 \tag{4.10}$$

$$2^{\frac{s_{i_{\max}}}{NN}} = \min(N_x, N_y) \quad \rightsquigarrow \quad s_i \leq NN \log_2 \left(\min(N_x, N_y)\right), \tag{4.11}$$

where $i = x, y$. But to take all the range described here would not be efficient in the sense that atoms with bigger scales are never chosen. Hence, the range is reduced and a choice that lead to good results is:

$$s_i \in [0, NN \cdot (\lfloor \log_2(N) \rfloor - 3)], \tag{4.12}$$

where the factor 3 has been added in order to not take too big scales (that almost never will be used).

Rotation As previously said, the rotation is in steps of 10^0 degrees. The phase, in order to cover all the spectrum, has to go from 0 to π radiants (because all the atoms are real and have a spectral component in the positive frequency plain and its symmetric component in the negative plain. Otherwise, it should move from 0 to 2π). So,

$$\text{phase}_n = n\Delta\theta = n\frac{\pi}{18}. \tag{4.13}$$

Size of the dictionary With the former assumptions, the size of the dictionary can be calculated: 26.836.992 atoms for an image of 128 squared pixels.

4.3.3 Dictionary $\mathcal{D}_G$ based on Gaussian atoms

The use of a dictionary based on gaussian functions is fully justified with the aim to code the baseband of the image. The AR atoms are suitable for coding contours, but they do not code the low frequency band of the image. In previous works ([31],[32]), the lower frequencies of an image were coded by either using a downsampled copy of the image or do not even code it, regarding this work to the atoms of the dictionary itself. The drawback of using a downsampled image to code the baseband is that it introduces artifacts due to the downsampling-upsampling process, then baseband coding could be greatly improved by adding a new dictionary: a gaussian dictionary, introduced by [38]. Also, an attractive feature is that dictionary will be pretty tiny and will not represent a load for the final coding process.

A Gaussian atom is defined as:

$$g_\gamma(x,y) = \sqrt{2}Ke^{-(x^2+y^2)}, \tag{4.14}$$

where γ is the set of parameters to generate the dictionary and K a constant that normalises the discrete norm of g_γ. Assuming our gaussian atoms equiscaled (so that is with a circular projection) we can define our parameters x_γ and y_γ:

$$x_\tau = \frac{(x-p_x)}{\sigma_x} = \frac{(x-p_x)}{2^{\frac{s_x}{NN}}} \tag{4.15}$$

$$y_\tau = \frac{(y-p_y)}{\sigma_y} = \frac{(y-p_y)}{2^{\frac{s_y}{NN}}}. \tag{4.16}$$

where $s_x = s_y = s$ because we choose isotropic gaussians. The design of the dictionary must be carefully done in order to cover all the baseband area in the Fourier domain; hence, to be dense in this frame of the Hilbert space. So, we must impose few constrains on the s parameter but, for simplicity, let's call $2^{\frac{s}{NN}} = \sigma_s$. We call σ_s due that this parameter refers to the *spatial* σ in contraposition with the *frequencial* σ_f. We have to review that the Fourier transform of a Gaussian is also a Gaussian and their σ's are connected by:

$$\sigma_s \cdot \sigma_f = 2. \tag{4.17}$$

Applying few constraints we might find the upper and lower bounds of σ_s:

- $\sigma_{s_{\max}} \rightsquigarrow$ The biggest Gaussian we will be able to put in our image (in spatial sense) will be determined by the image itself: its size (taking the smallest side of an $N_x \times N_y$ pixels image). We know that the 99% of the energy of the gaussian is concentred in the $6\sigma_s$ area of it. This leads us to:

$$6 \cdot \sigma_{s_{\max}} = \min(N_x, N_y) \rightsquigarrow \sigma_{s_{\max}} = \frac{\min(N_x, N_y)}{6}. \tag{4.18}$$

  This Gaussian will have the lowest frequency representation, so by appliying 4.17 we find the smallest bandwidth our Gaussians will cover:

$$BW_{\min} = \sigma_f \cdot 3 = \frac{2}{\sigma_s} \cdot 3 = \frac{36}{\min(N_x, N_y)}. \tag{4.19}$$

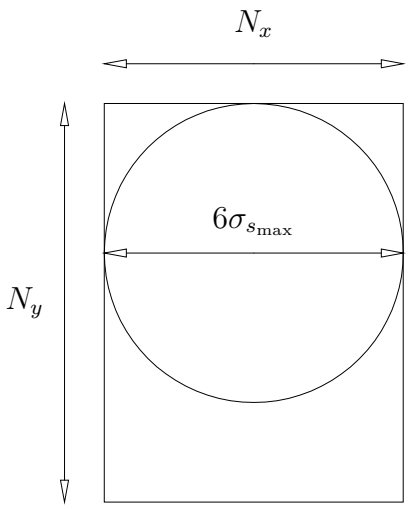


Figure 4.5: Framework to calculate $\sigma_{s_{\max}}$

- $\sigma_{s_{\min}} \rightsquigarrow$ The previous constrain was found in the spatial domain of the image; this other is found in the Fourier domain. As we want to cover the baseband of the Fourier domain, we have already defined the constrain that will give the lowest frequency; to find the upper bound we have to find the point where our gaussians collide with the lowest central frequency of the AR dictionary atoms. We have to find the frequency (ω_x, ω_y) of the atom with lowest frequency where it have maximum power. First, let's find the Fourier transform of a generic AR atom:

$$G_\lambda(\omega_x, \omega_y) = -\pi\sigma_x\omega_x^2 e^{-\frac{(\sigma_x\omega_x)^2+(\sigma_y\omega_y)^2}{4}}, \tag{4.20}$$

where $\sigma_x = 2^{\frac{s_x}{NN}}$ and $\sigma_y = 2^{\frac{s_y}{NN}}$ are the scaling factors of the anisotropical refined Gabor atom as we defined in 4.6 and 4.7. To find the maximum of this function we can impose the condition $\omega_y = 0$ due to the simetry of our spectrum. So, we have:

$$\frac{\partial G_\lambda(\omega_x, 0)}{\partial \omega_x} = -2\pi\sigma_x\omega_x e^{-\frac{(\sigma_x\omega_x)^2+(\sigma_y\omega_y)^2}{4}} \left(1 - \frac{(\omega_x\sigma_x)^2}{4}\right) = 0. \tag{4.21}$$

Avoiding the trivial solution $\omega_x = 0$, it leads to find the lowest frequency $\omega_{x_{\min}}$ of the AR atoms (so that is when the scaling σ_x factor is maximum):

$$\omega_{x_{\min}} = \frac{2}{\sigma_{x_{\max}}}. \tag{4.22}$$

Also this lowest frequency of the Gabor atoms set is the biggest bandwidth our Gaussian dictionary will represent, that is

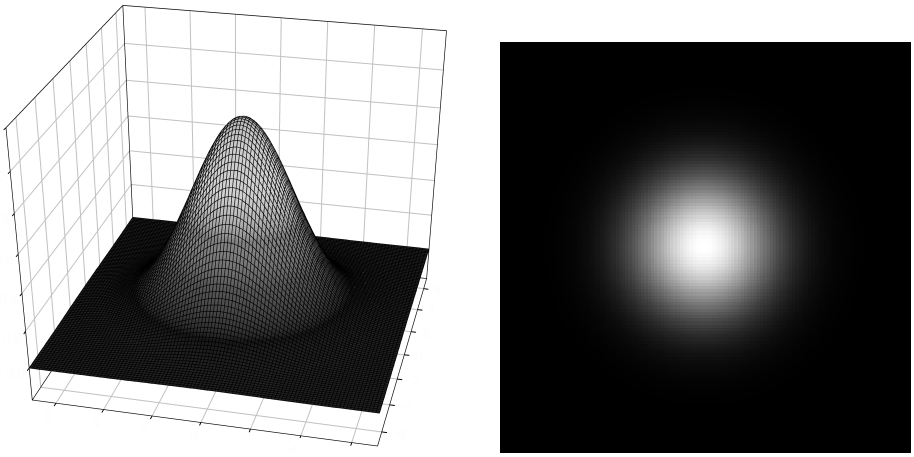
$$BW_{\max} = \frac{2}{\sigma_{x_{\max}}}. \tag{4.23}$$

Once we have set this lowest frequency, we can find the lower bound of σ_s. Taking in consideration:

$$3\sigma_f = \frac{2}{\sigma_{x_{\max}}}. \tag{4.24}$$

Then, by applying 4.17 we find:

$$\sigma_{s_{\min}} = 3 \cdot \sigma_{x_{\max}}. \tag{4.25}$$



(a) (b)

Figure 4.6: Gaussian atom.

4.3.4 Our choice

The design parameters chosen for this Gaussian dictionary are:

Translation We will make the same assumptions done in the previous dictionary.

Scales By strictly applying the relations 4.18 and 4.25 we find that (in concrete for our squared images of 128 pixels):

$$\sigma_{s_{\max}} = \frac{128}{6} \approx 21 \tag{4.26}$$

$$\sigma_{s_{\min}} = 3\sigma_{\mathrm{AR}_{\max}} = 3 \cdot 2^{\frac{12}{3}} = 48. \tag{4.27}$$

The reader can see that these bound for σ_s are not logical in the sense that $\sigma_{s_{\min}} > \sigma_{s_{\max}}$. The reason is because our Anisotropic Refinement dictionary has too big

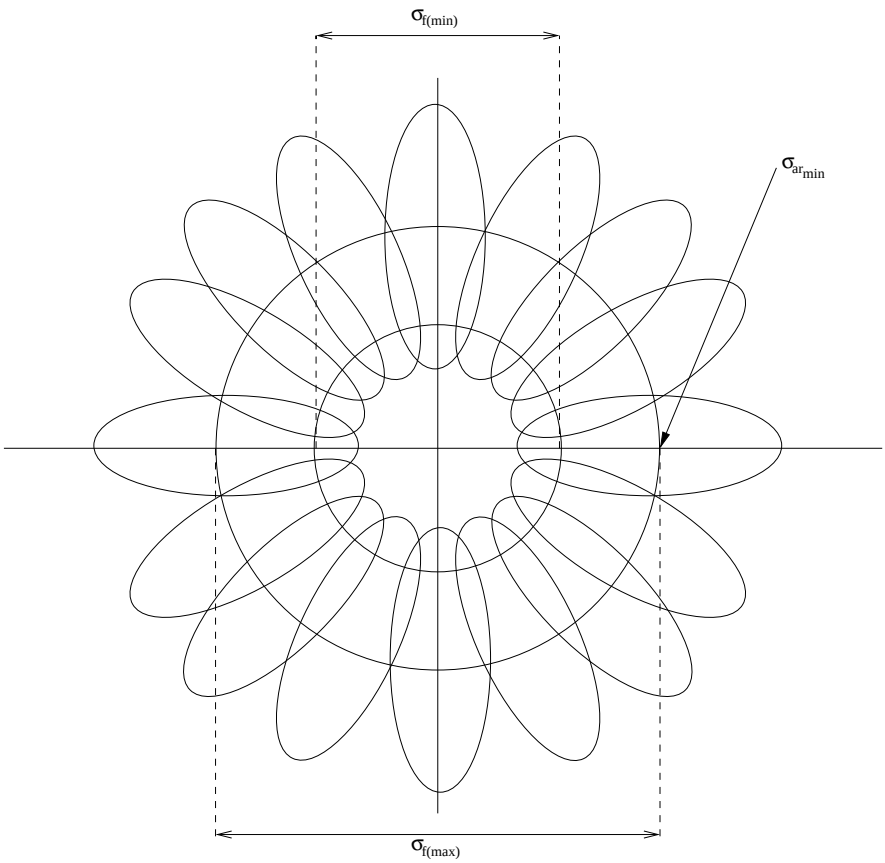


Figure 4.7: Gaussian design frequency scheme. The bigger circle corresponds to the spectrum of the Gaussian with $\sigma_{s_{\min}}$ (and $\sigma_{f_{\max}}$ inversely) defined by the lowest frequency of the AR Atoms. The small circle corresponds to the $\sigma_{s_{\min}}$ (and $\sigma_{f_{\min}}$ accordingly) defined by the size of the image.

scales. There would be two solutions: to reduce the scales of the AR dictionary or relax our assumptions in the calculation of $\sigma_{s_{\min}}$. The first option leads to reduce the AR dictionary and the following results are poor because Matching Pursuit does not have enough atoms to catch efficiently the structures of the image. The second option seems more feasible, thus, we will relax the constrain and, instead of taking the maximum $\sigma_{\mathrm{AR}_{\max}}$ we would take the first σ'_{AR} that accomplishes the criteria. Hence,

$$\sigma_{s_{\min}} = 3\sigma'_{\mathrm{AR}} = 3 \cdot 2^{\frac{8}{3}} \approx 19, \tag{4.28}$$

and, finally, $\sigma_s \in [19, 21]$.

Size of the dictionary With the former assumptions, the size of the dictionary can be calculated: 49.152 atoms for an image of 128 squared pixels.

4.3.5 Dictionary $\mathcal{D}_{Ga}$ based on Gabor Atoms

In most of the previous research done on Matching Pursuit over images, most of the decompositions have been done over dictionaries based on Gabor [40] atoms due to its good time-frequency localization. For our purposes, the use of Gabor atoms is justified due to the fact that they are appropriate functions to code patterns and textures.

A Gabor atom is defined as a modulated Gaussian function:

$$g_\gamma(x,y) = \sqrt{2}Ke^{-(x^2+y^2)}e^{i(\omega_x x+\omega_y y)}. \tag{4.29}$$

But, for our purposes, we will take just the real part of $g_\gamma(x,y)$. Then we will apply the rotations we will generate the necessary frequencies in the y axis:

$$g_\gamma(x,y) = \sqrt{2}Ke^{-(x^2+y^2)}\cos(\omega_x x), \tag{4.30}$$

where $\gamma = (\boldsymbol{p}, s, \theta, \omega_x)$ defines the set of parameters in the same way we did to create the other dictionaries. Note that we will take isotropic Gaussians instead of anisotropic Gaussians here. The only new constrain here is the election of the frequencies ω_x; we must choose the frequencies carefully to acomplish the Nyquist criteria and set an overlap ratio that would guaratee the density condition in the associated Hilbert space.

Then for each scaling factor $\sigma_s = 2^{\frac{s}{NN}}$ we must calculate the set of frequencies ω_{x_i} it will represent. For our purpose we can define a set of constrains that will determinate those frequencies:

Condition 1: It is clear that a determinated Gaussian will be able to represent a range of (normalized) frequencies $[\omega_{x_{\min}}, 1]$ where $\omega_{x_{\min}}$ is the minimum frequency. This minimum frequency will the determinated by the size of the Gaussian, so:

$$6\sigma_s > T \rightsquigarrow \omega_x > \frac{1}{6\sigma_s}, \tag{4.31}$$

where T is the spatial period of the atoms. We can rewrite this equation in term of the scaling factor s:

$$\omega_x > \frac{2^{-\frac{s}{NN}}}{6}. \tag{4.32}$$

Also, sometimes is not so necessary to reach the lower bound due the low frequencies would be already coded by either the Anisotropic refinement atoms or the Gaussian atoms. So, it is possible to relax the lower frequency condition by appliying a *correction factor*:

$$6\sigma_s > \alpha T \rightsquigarrow \omega_x > \frac{\alpha}{6\sigma_s}. \tag{4.33}$$

Then the corrected lower frequency is:

$$\omega_x > \alpha\frac{2^{-\frac{s}{NN}}}{6}. \tag{4.34}$$

Remember that we are here working with normalized frequencies. If we want to work with absolute frequencies we have to multiply by the sampling frequency, that is $\frac{N_x}{2}$.

Condition 2: Once we have set the range of our frequencies, we must define the step between two represented frequencies in this set. That is the increment between two adjacent frequencies. We could represent *all* the frequencies in the range $[\omega_{x_{\min}}, 1]$ incrementing by unary steps $\frac{2}{N_x}$. But this would be a waste of resources due that it is not necessary because, according with the Figure 4.8, we would do too much overlapping. We can overlap less, for example just half a Gaussian in the Fourier domain:

$$\Delta f = 3\sigma_f. \tag{4.35}$$

Or, rewritten into terms of s:

$$\Delta f = 6 \cdot 2^{-\frac{s}{NN}}. \tag{4.36}$$

Also, we can control the overlap:

$$\Delta f' = \beta \Delta f \qquad \beta \in (1, 2]. \tag{4.37}$$

Then, finally, we can generate the associated frequencies per each scaling factor s:

$$\omega_{x_i}(s, NN) = 1 - i\Delta f'(s, NN) \tag{4.38}$$

$$i \in [0, j \text{ t.q } \omega_{x_j} > \omega_{x_{\min}}]. \tag{4.39}$$

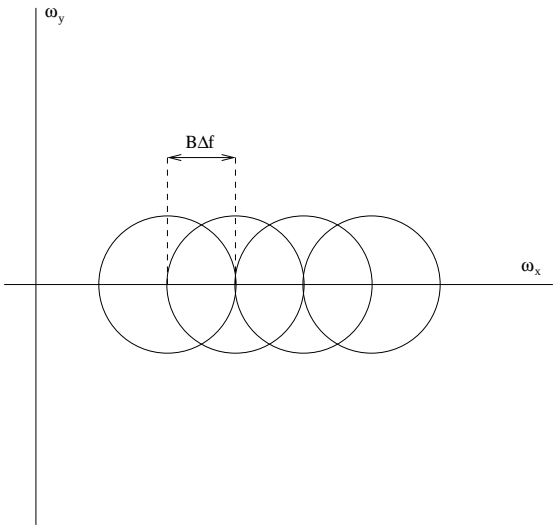
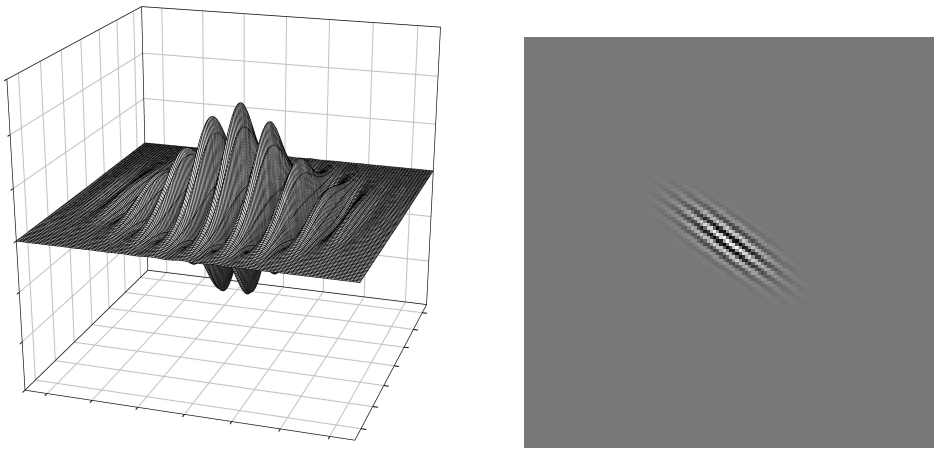


Figure 4.8: Incremental factor.

4.3.6 Our choice

The design parameters chosen for the Gabor dictionary are:

Translation We will make the same assumptions done in the previous dictionaries.



(a) (b)

Figure 4.9: Gabor atom.

Scaling A good choice for the scaling factor for our isotropic atoms, based on empirically results, is:

$$s_i = \left[\frac{NN \cdot (\lfloor \log_2(N) \rfloor - 3)}{2}, NN \cdot (\lfloor \log_2(N) \rfloor - 3) \right] \tag{4.40}$$

Correction parameters The adjustable parameters that defines our dictionary have been set as:

$$\alpha = 1.5 \qquad \beta = 1 \tag{4.41}$$

Size of the dictionary With the former assumptions, the size of the dictionary can be calculated: 44.236.800 atoms for an image of 128 squared pixels.

Part II

Results

Chapter 5

Results

5.1 Introduction

The standard flavor of Matching Pursuit defined in Chapter 3 reconstructs an image by placing atoms from a redundant dictionary in order to catch the maximum of the energy from the residual image $R^n f$ at each iteration[1].

But the problem that naturally arises is: is the decomposition given by Matching Pursuit using these two dictionaries efficient in the sense that Anisotropic Refinement atoms will always be destined to code edges and Gabor atoms to code textures? The viceversa case would lead to non-efficient decompositions (i.e. code a texture formed by squares by placing a lot of Anisotropic Refinement atoms instead of use few Gabor atoms). To avoid these situations, our **Hybrid Matching Pursuit** coding scheme would include a mechanism based on edge and texture probability masks to *help* the standard Matching Pursuit algorithm to place efficiently the atoms from one or the other dictionary to code a determinate feature. Furthermore, other schemes will be presented in order to improve the decompositions given by the standard Matching Pursuit algorithm as the **Scrambled Matching Pursuit** and the **Split Matching Pursuit**.

Another issue of Matching Pursuit is whether the atoms it is placing are the best ones according to the Human Visual System (HVS), in other words, is the atom that catches the maximum of the energy at the n-th iteration the one that better catches the structures our eyes perceive? At this point we have to recall that our dictionary based on Anisotropic Refinement atoms matches the responses of the early visual cortex cells [72]. Then, it seems feasible that Matching Pursuit performs also properly in terms of the HVS. That means that the decompositions given by Matching Pursuit should be appropriate in terms of visual quality due to this correlation between our dictionary and the perception process.

To check the former assumption, we would design a Matching Pursuit coder that would place more atoms where there is more visual information (such as strong edges and patterns). Though, an hybrid version of Matching Pursuit aided by the information given by the probability masks (edge or textures mask or both at the same time) would lead

[1]Here the name of *greedy approximation*

to decompositions that will be more affine to how the human people perceive the images (even if the PSNR ratio is lower). At this point, we will compare the results given by the standard Matching Pursuit decomposition an the decompositions given by our hybrid schemes.

Prior to any description of the the fully functional Hybrid Matching Pursuit coder aided by probability masks, some definitions and constraints must be introduced. First some theoretical definitions and comments about the probability masks: definition and properties. After that, the description of the Hybrid Matching Pursuit coder will be introduced and the other coding schemes cited above as well. Finally, some results about the performance of these algorithms will be shown.

5.2 Probability masks theoretical description

In Chapter 2, few edge detectors were presented and the contrast based detector was chosen for our purposes due to its property to be able to detect edges on high contrast images. Moreover, texture detectors were introduced and Simoncelli's one [84] was chosen for its performance and capacity to deal with textures present in natural images. Furthermore, Simoncelli's texture detector is very suitable for our purposes for its natural definition based on the human perception of textures (concretely, in the response of the V1 cells [72]). These two detectors give an important information that will be used to define the probability masks necessary for the design of our Hybrid MP coder. Let us define, prior to make a formal description of the coder, some properties and definitions on the probability masks.

5.2.1 Definition

Strictly speaking, a probability mask can be defined as a matrix

$$\mathcal{P}(x,y) \in [0,1] \qquad (x,y) \in [0, N_x - 1] \times [0, N_y - 1], \tag{5.1}$$

where each element (x, y) has the probability value (limited between 0 and 1) to have a determinate feature (in our case textures or edges) at this position of the original image. There could be many ways to define probability masks for textures or edges based on the results given by the texture and edge detectors. Nevertheless, this definition must be done taking into account how Matching Pursuit makes an atomic decomposition of an image in order to have the probability masks matched with it.

In the texture case, it is easy to compute the probability mask because we can not model how the atoms are placed by Matching Pursuit to shape a texture (recall here that the atoms that will, mainly, represent a texture will be Gabor atoms). Hence, the texture probability mask, let us call it $\mathcal{P}_T(x, y)$, would be defined as

$$\mathcal{P}_T(x,y) = \Psi[T(x,y)], \tag{5.2}$$

where $T(x, y)$ is the texture detector and $\Psi[I(x, y)]$ is a linear mapping application defined by

$$\Psi[I(x,y)] = \frac{I(x,y) - \min\{I(x,y)\}}{\max\{I(x,y)\} - \min\{I(x,y)\}}. \tag{5.3}$$

In fact, the function of this mapping application is to re-scale the original image $I(x, y)$ into the interval $[0, 1]$ necessary to accomplish the constrains of Eq. 5.1.

In the other case, the edge probability mask, we can not define it directly as the result given by the contrast edge detector because when an edge is shaped by Matching Pursuit, the center of the atoms are not placed in the position of the edge itself. Even, in the simplest case, when an edge is modelled by two atoms, the positions of these atoms are near the edge, but not on the edge (see Figure 5.1). So, a model for the atom placement must be defined according to the set of parameters γ that defines an atom. With the combination of this model and the edge detector results, we could define a probability mask for the placement of Anisotropic Refinement atoms to shape an edge. This model will be defined as a linear filter that will enlarge the area around an edge to cover the positions near it, where a high probability to place an atom exists. Once this filter is defined, we can find the edge probability mask $\mathcal{P}_E(x, y)$ as

$$\mathcal{P}_E(x,y) = \Psi\left[E\left(x,y\right) * \varphi\left(x,y\right)\right], \tag{5.4}$$

where $\varphi(x, y)$ is the edge probability placement filter expressed in cartesian coordinates and $*$ the linear convolution operator.

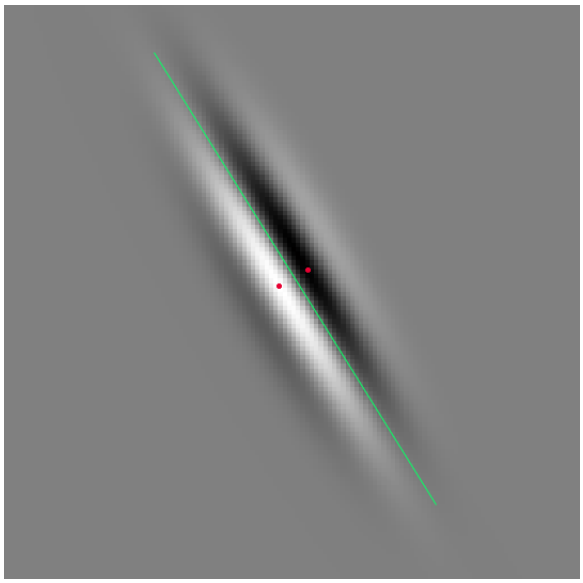


Figure 5.1: Edge representation by two Anisotropic Refinement atoms. When Matching Pursuit represents an edge, assuming a simple model of two atoms, the result looks like this. The green line represents the edge and the two red spots the center of the two atoms. This figure shows that to shape an edge, the atoms are not on the edge itself but near it.

5.2.2 Edge model

First to define a edge probability mask, a careful study over the behavior of Matching Pursuit in front of an edge must be done, so, how the edge is represented by a combination of atoms. Once, this model is set, an edge probability mask can be defined according to it, in the sense that the areas where there is a higher probability to place an atom must be enhanced.

To do the mathematical analysis a bit more simpler (but still valid) we will represent an edge in its 1D version. Let us represent an edge as a discontinuity in the intensity of an image then, its 1D version, could be modelled as a Heaviside function

$$\mathcal{I}(x) = \begin{cases} I_{\min}, & \text{if } x < 0 \\ I_{\max}, & \text{if } x > 0 \end{cases} . \tag{5.5}$$

As discussed on Chapter 4, the edges are represented by Anisotropic Refinement atoms hence, the 1D Anisotropic Refinement atom version is

$$g_\gamma(x) = \left(4x^2 - 2\right) e^{-x^2}, \tag{5.6}$$

with its set of 1D parameters $\gamma = (p_x, s_x)$. However, it can be rewritten in a handier expression for our purposes

$$g(x, s_x) = \left[4\left(\frac{x}{s_x}\right)^2 - 2\right] e^{-\left(\frac{x}{s_x}\right)^2}. \tag{5.7}$$

Then, we have to define how to model an edge by a linear combination of AR atoms and the simplest representation that could be done is by taking two atoms as represented in Figure 5.2. According with this figure, the analytical form of this approximation is

$$\tilde{\mathcal{I}}(x) = K_1 \cdot g(x + c_1, s_{x_1}) - K_2 \cdot g(x - c_2, s_{x_2}), \tag{5.8}$$

where c_1 are c_2 the center of each atom, s_{x_1} and s_{x_2} the scaling factors[2] and K_1 and K_2 the amplitude constants to shape the approximation of the edge. To define our edge probability masks the only parameter we have to take into account is the center of the atoms because they will determinate the area where an atom can be placed. The atoms that best shape out edge model have its first zero crossing in the coordinates origin. If we compute analytically this values, we find

$$c_1 = c_2 = \frac{\sqrt{2}}{2} s_x \tag{5.9}$$

[2]These scaling factors, and even the center parameters and the shaping constants, must not be the same and their values depend on several factors as the inclination uniformity of the step, its symmetry,... In the particular case depicted in Figure 5.2 they coincide because the edge has uniform inclination. We will assume that all our edges accomplish this property.

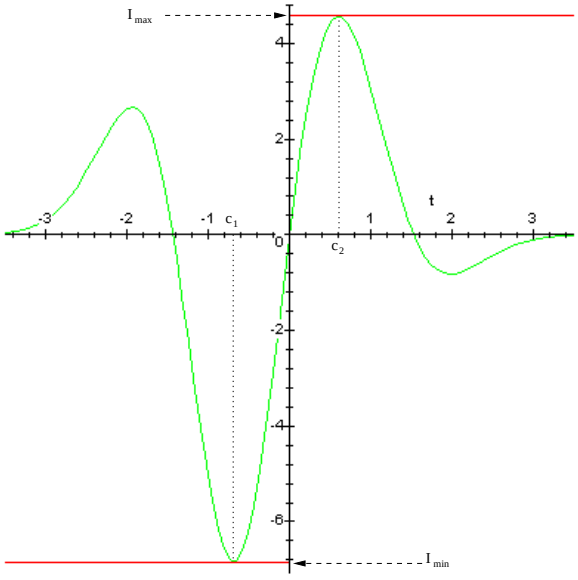


Figure 5.2: Edge Model. The red plot depicts an edge while the greed represents the simplest representation of an edge by taking the linear combination of two Anisotropic Refinement atoms.

The next step is to define the area around an edge where there is a high probability to place an atom according to our model and assumptions. This probability area will exclusively depend on the scales of the atoms, so we can define this probability area as

$$\delta = \left[\frac{\sqrt{2}}{2} s_{x_{\min}}, \frac{\sqrt{2}}{2} s_{x_{\max}} \right]. \tag{5.10}$$

Finally, we can define a linear filter that will take into account all this information to refine the result given by the edge detector by placing a high probability to have an atom in the areas around the detected edges. Let us define the edge probability placement filter $\varphi(r, \theta)$ expressed in polar coordinates as

$$\varphi(r, \theta) = \begin{cases} 1, & \text{if } r < \delta_{\max} \\ e^{-\left(\frac{r-\delta_{\max}}{\sigma}\right)^2}, & \text{if } r > \delta_{\max} \end{cases} \tag{5.11}$$

where σ is the roll-off parameter.

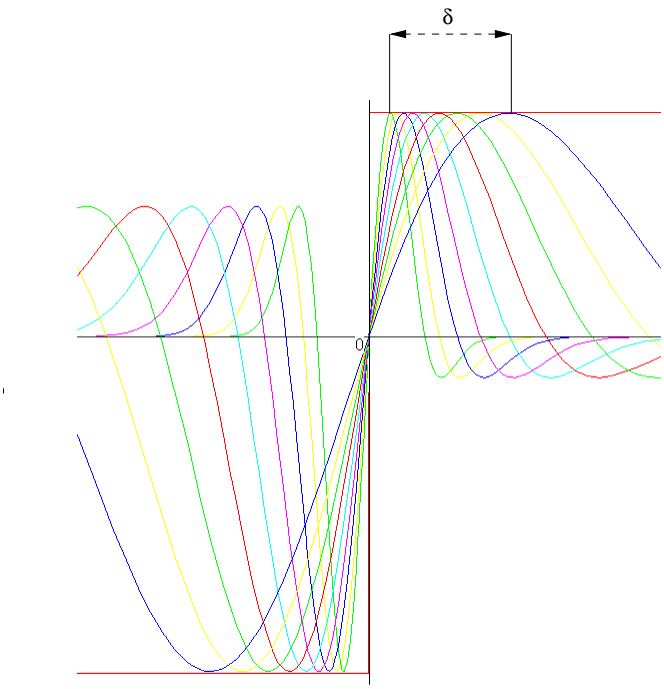
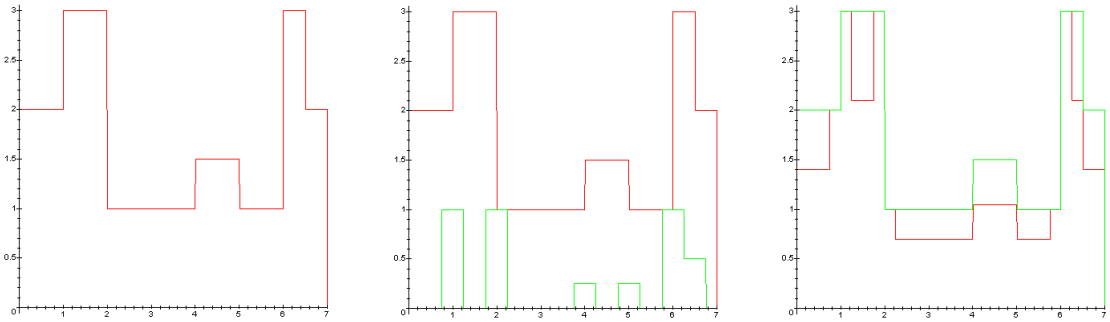


Figure 5.3: Atom placement area δ.



(a) (b) (c)

Figure 5.4: Masking process example. Figure (a) depicts the original 1D signal, (b) the original (red) and the mask (green) and (c) the original (green) and the masked signal (red). Observe how the areas around the discontinuities (edges of the image) have been preserved due to the high probability to place atom.

5.2.3 Mutual masks

Once the models for the two types of masks are defined a further refinement could be done: use the information of one mask to accurate the other. In our particular case of edge and texture masks, if there is an area with a high probability to be a texture, this should be taken into account into the edge mask and decrease the probability of edge in that area.

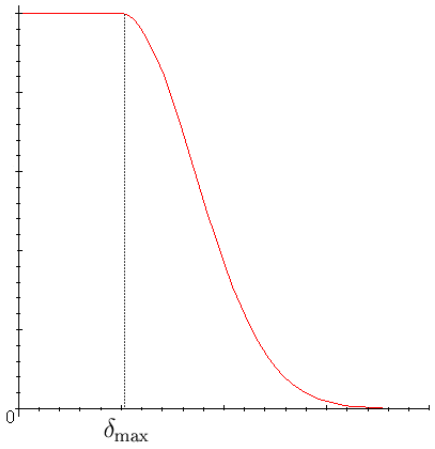


Figure 5.5: Edge model.

With this method, a better probability exclusion over the two masks is achieved. Hence, the final masks would be defined as:

$$\mathcal{P}_E(x,y) = \Psi\left[E\left(x,y\right) * \varphi\left(x,y\right)\right] - \gamma_1 \cdot \Psi[T(x,y)] \quad (5.12)$$

$$\mathcal{P}_T(x,y) = \Psi[T(x,y)] - \gamma_2 \cdot \Psi\left[E\left(x,y\right) * \varphi\left(x,y\right)\right], \quad (5.13)$$

with the *mutual influence parameters* $\{\gamma_1, \gamma_2\} \in [0,1]$ and we will assume $\gamma_1 = \gamma_2$ to achieve an equal mutual influence. Empirically, the results obtained were good for ranges of $\gamma \in [0.3, 0.5]$.

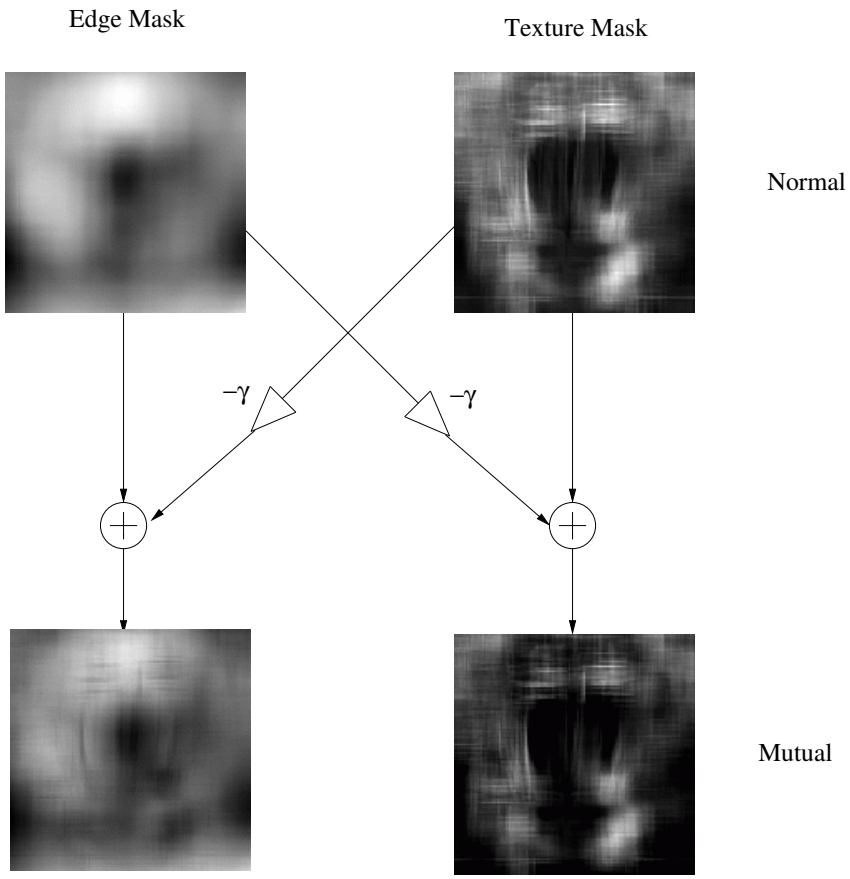


Figure 5.6: Mutual masks definition.

5.3 Hybrid Coder description

5.3.1 Introduction

Once we have defined a model for the probability masks a general Hybrid Matching Pursuit coder can be defined as depicted in Figure 5.7. Then the question is: why do we use probability masks? The aim of this coder is to obtain a decomposition of the target image in the same way Matching Pursuit standard algorithm does but enhancing the areas with more visual information. Actually, this idea is not new: use masks to enhance areas in the target image when the coding is performed has been already used in [10]. It is true that the standard Matching Pursuit does not take into account where are the most meaningful areas of the image. Thus, our probability masks enhance those areas where are more visual information: edges and textures. It is also true that this decomposition will not give better PSNR results in comparison with the standard Matching Pursuit but, in some cases, will achieve better visual quality (according with some subjective tests).

Here there are commented in detail the operation steps of the coder:

- A Full Search Matching Pursuit is done at the n-th iteration by taking the residual image (the original after subtracting the previously selected atoms).

- Once we have performed the convolution in the frequency domain, matrix $\boldsymbol{A}$ corresponding to all the scalar products of an atom (with all its displacements all over the image) is obtained. Then, the standard Matching Pursuit algorithm would choose the atom that presents the largest scalar product. Instead of that, a decision matrix $\boldsymbol{A}'$ is computed as:

$$\boldsymbol{A}' = \boldsymbol{A} \odot \mathcal{P} \tag{5.14}$$

  where $\odot$ is the term-by-term product operator and $\mathcal{P}$ is a probability mask. That means that each position of the atom is pondered by the probability to have this feature in this point. The election of which probability mask $\mathcal{P}$ is used depends on the atom evaluated: edge probability mask if it is an Anisotropic Refinement atom or texture probability mask if it is a Gabor atom. Then the election of the strongest atoms is taken from $\boldsymbol{A}'$, but the reconstruction and updating of the residual part is done with the original scalar product to keep energy conservation.

- An optional step in our coder would be to update the probability masks with the data of the atom chosen. In further sections, this point will be analyzed thoroughly and few results shown.

- Finally, the last step consist into the quantization and codification of the Matching Pursuit stream (that is the coefficients obtained for each atom c_n and the intrinsic parameters of the atom γ_n).

The decoding operation is extremely simple (a good feature for a decoder, indeed): generate the atoms and place them in its position.

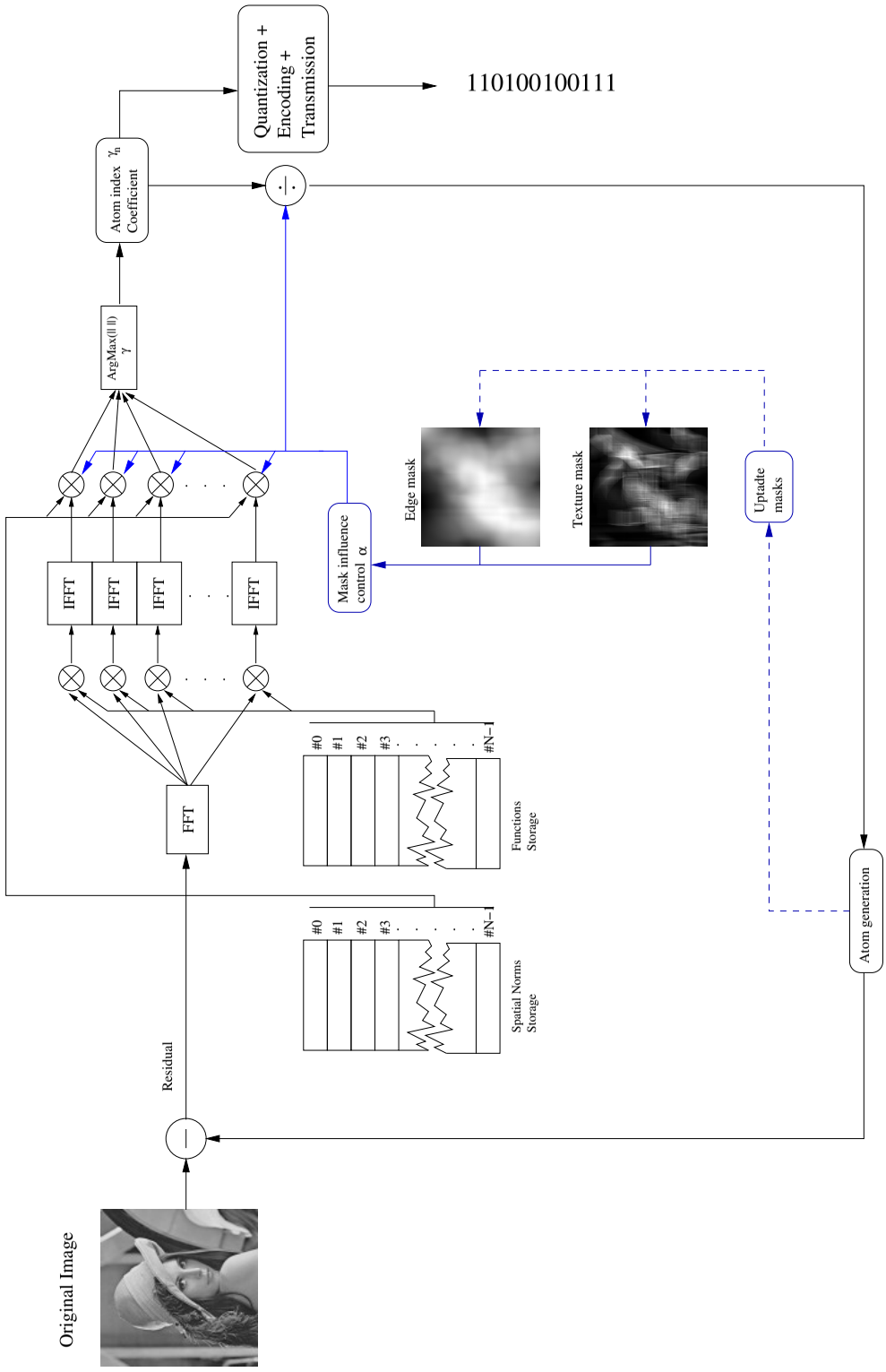
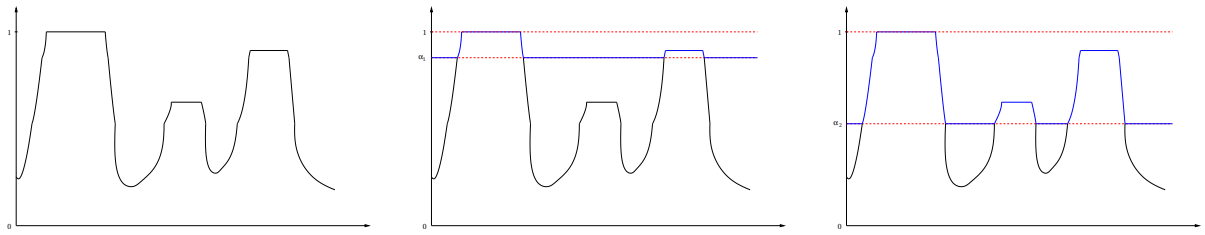


Figure 5.7: Hybrid Matching Pursuit Coder.

5.3.2 Mask control α: Waterfilling

In the previous section, the description of both probability masks for textures (shaped by Gabor atoms) and edges (shaped by Anisotropic Refinement atoms) have been introduced. Once arrived to this point, we have to introduce a new parameter to the description of our hybrid coder: the possibility to control how much influence these masks will have in our Matching Pursuit algorithm.



(a) (b) (c)

Figure 5.8: Waterfilling. The α parameter allows to control the influence of the masks into the ponderation process. In (a), the original a 1D-Mask. In (b), its waterfilled result with a high value of α and, in (c), with a lower value. Note that in (c) begins to appear a weak probability region (lower peak).

If we recall Chapter 3, Matching Pursuit decomposition is performed by using the full-search algorithm in the frequency domain. According with this full-search scheme, the largest scalar product is chosen after apply the IFFT to obtain the final scalar product matrix corresponding to all the scalar products of an atom displaced around all the spatial positions. At this point is where the probability masks is applied to ponder the scalar products obtained for each position. Once all the largest ponderated scalar product is computed for each, the maximum of all them is selected and the corresponding atom is chosen. If the masks (either edges or textures) are applied directly, the obtained results are extremely poor (in both terms of PSNR or HVS). For example, one of the drawbacks of the ponderation by probability masks is that in the places where it reaches a zero level, the scalar product will be also zero and there will never be placed any atom. All types of probability masks are bounded into the interval $[0, 1]$ due to the $\Psi[I(x, y)]$ operator but this wide range introduces these insidious effects on the whole system hence, a control over the effect of the masks must be introduced.

The ponderation effect, so, how enhanced are certain areas of the scalar product matrix $\boldsymbol{A}$, is determined mainly by the value range of the probability mask (by definition, the probability masks $\mathcal{P}_E(x, y)$ and $\mathcal{P}_T(x, y)$ ranges into $[0, 1]$). Let us call α to the lower bound of a probability mask $\mathcal{P}$:

$$\alpha = \min\{\mathcal{P}\}. \tag{5.15}$$

Then, the influence of the probability mask on the scalar product matrix $\boldsymbol{A}'$ defined by Eq.5.14 will decrease as $\alpha \to 1$. It happens due to that the poderation is done over an

interval of size $1-\alpha$, hence the effects of the ponderation will be more noticeable as wider this interval will be. But the problem is: how can be the a probability mask bounded in the interval $[\alpha, 1]$? The mask should be bounded in such way that the wider the interval, the lighter the influence of the mask. A good strategy would be to use a waterfilling technique. That is to generate the ponderation mask in the following way:

$$\mathcal{P}_\alpha(x,y) = \begin{cases} \mathcal{P}(x,y), & \text{if } \mathcal{P}(x,y) > \alpha \\ \alpha, & \text{otherwise} \end{cases} . \tag{5.16}$$

The procedure to create a probability mask is shown in the Figure 5.8. With this technique two objectives are achieved: to control the influence of the masks to avoid over-ponderation and to enhance gradually the influence areas as α increases. When α take a high value (so the ponderation range is small) only the areas with a high probability are emphasized over the whole mask (subfigure (b) of Figure 5.8). As alpha decreases, the other areas with a smaller value appear and are taken into account to perform the ponderation (subfigure (c) of Figure 5.8).

Finally, it must be remarked that the election of the value of α is chosen empirically. As we will comment in the conclusions, the use of probability masks when performing a Matching Pursuit decomposition is an ill-posed problem. The results are extremely depending on the image itself, hence to make an analytic study is too complicated.

5.3.3 Masks updating

An issue that we tried to embed in our Hybrid Matching Pursuit coder was the possibility to update the probability masks. It was done in order to take into account the information that was extracted with the subtraction of an atom at the n-th iteration. Actually, it is a good idea because when the algorithm has taken out few atoms, the masks that are applied are still the ones computed for the original image. Two techniques were proposed for the edge detector: to subtract a pondered version of the atom's envelope to the masks or to design a linear edge estimator (basically the Sobel edge detector over a smoothed version of the original image) and update it accordingly. Both proposals were tested and, despite it seemed to be a logical assumption to update the masks, the results were, though, poor. Nevertheless, the update based on the linear edge estimator performed a bit better and the updated version of the probability mask (for edges) looked curious (see Figure 5.9). See also Future Work Section.

5.3.4 Quantification and Codifications

The Matching Pursuit output consists of:

- The dictionary to whom the atom belongs: $\mathcal{D}_{AR}$, $\mathcal{D}_G$ or $\mathcal{D}_{Ga}$ (Anisotropic Refinement, Gaussian or Gabor atoms respectively).

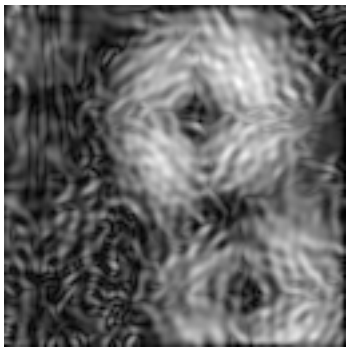


Figure 5.9: Updated edge probability mask for LENA image.

- The set of parameters that defines the shape of the atom γ_n. Note that this parameters (position, orientation, scale(s), frequency (in the case of Gabor)) are all integer.
- The coefficient c_n of each atom. Note that in this case the coefficient is a real value.

Then, the only parameter that needs to be quantified is the coefficient c_n [31]. Regarding the quantification matter, many techniques have been purposed: uniform quantization [31], death zone quantization [69],... But the one with best performance is described in [37]: it takes advantage of the exponential decay of the coefficients and optimizes the quantization. This scheme consists on a posteriori non-uniform quantization, that means that the encoder does not use the quantized coefficients to update the residual[3]. The a posteriori scheme is practical for asymmetric systems: the Matching Pursuit stream is computed once and the quantized several times to satisfy different rate constraints.

Just few words about this scheme (for a detailed description on the quantizer, check [37]). The coefficient energy decreases with the iteration number and it can be upper-bounded by an exponential decay curve (see Figure 5.10) which only depends on the properties of the dictionary and the search algorithm

$$|c_n| \leq \left(1 - \alpha^2\beta^2\right)^{\frac{n}{2}} \|f\|, \tag{5.17}$$

where β is a redundancy factor and $\alpha \in (0, 1]$ represents an optimally factor. Actually, α depends on the algorithm that, at each iteration, searches for the best atom in the dictionary; it is set to one when MP browses the complete dictionary at each iteration. The redundancy factor β depends on the dictionary construction[4]. These two factors are used to design the quantization scheme.

Let n_j be the number of quantization levels for the j^{th} coefficient. The Exponential Upper-bounded Quantizator (EUQ) assigns $n_{j+1} = (1 - \alpha^2\beta^2)^{\frac{1}{2}} n_j$ levels to the next coefficient $(j + 1)^{\text{th}}$. Bits are optimally distributed between successive coefficients accordingly

[3]The schemes that quantizes each coefficient while performing the decomposition in order to update the residual are called in-loop quantizator [60].

[4]Actually, β can be proportional to the coherence μ of the dictionary defined in Chapter

to their relative contribution to the signal representation.The quantization range and the number of quantization step are therefore reduced along the iteration number.

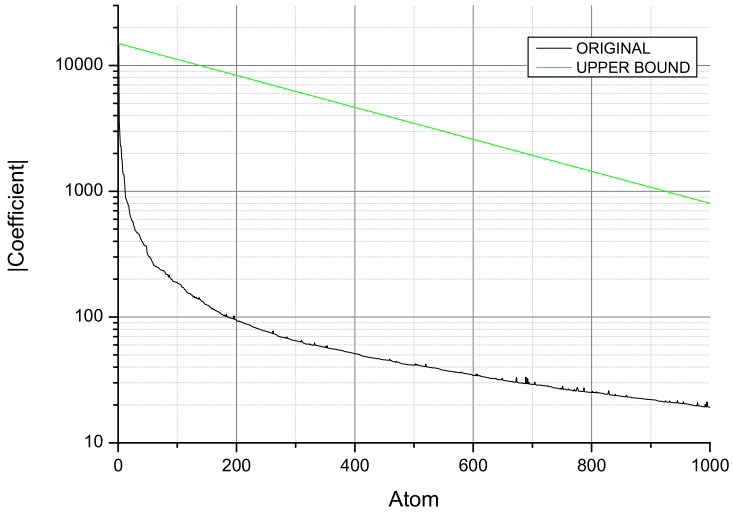


Figure 5.10: Coefficients norm upper-bounded with the exponential decay curve $(1-\alpha^2\beta^2)^{\frac{1}{2}}$ with $\alpha = 1$ and $\beta = 0.1$.

Finally, for the last step of the Matching Pursuit compression scheme, we use a classical arithmetic code. Arithmetic encoding [94] avoids the integral bit constraint of a Huffman code by coding together a sequence of symbols. Such a sequence is represented as an interval included in $[0, 1]$. The longer the sequence is, the smaller is the interval and the larger is the number of bits to specify the interval. The resulting bit rate is very close to the entropy and is generally better than with the Huffman coder.

Arithmetic coding works by using a probability interval defined with variables L and R, which are initially set to 0 and 1 respectively. The value of L represents the smallest binary value consistent with a code representing the symbols processed so far. The value of R represents the product of the probabilities of those symbols. To encode the next symbol, which is the j^{th} of the alphabet, both L and R must be recomputed as:

$$L = L + R\sum_{i=1}^{j-1} p_i \tag{5.18}$$

$$R = Rp_{j-1}. \tag{5.19}$$

This preserves the relationship between L, R and the symbols that have been previously processed. At the end of the message, any binary value between L and $L + R$ will unambiguously specify the input message. The interval will continue to be redefined until and end-of-sequence marker is coded. An example of a coding sequence can be found in Figure 5.11; it details the coding of the sequence of symbols IUI where each symbol has a

predefined probability. The top line is a scale. Each succeeding line represents a stage of the coding process. The first stage is the initial stage $[0, 1)$, which is followed by succeeding intervals. At each stage the current interval is the shaded rectangle. The value of L is the left hand side of the interval and the value of R is the length of the interval.

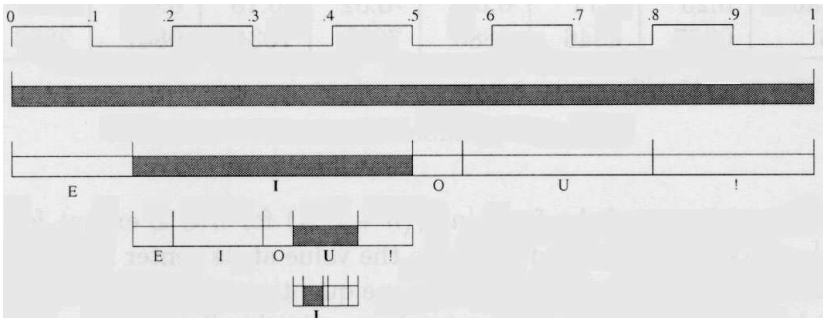


Figure 5.11: Arithmetic coding of the sequence IUI.

5.3.5 Hybrid Matching Pursuit coder aided by probability masks: Results

In this section some results are presented. First, the those related with the Matching Pursuit decomposition with one dictionary (AR atoms one) aided with an edge probability mask. Secondly, the results for the decomposition over the two dictionaries with both probability masks for edges and textures.

In relation with the quantification and coding of the resulting images, a deep study has been already done. These studies ([31],[37]) have exploited the characteristics (coefficients, atoms,...) of the Matching Pursuit coding algorithm and their results have been applied to design both the quantifier and the Arithmetic coder. Hence, in this study, comparative results on the coding are shown to have an idea the performance in terms of bit rate of each algorithm.

Single dictionary probability mask aided Matching Pursuit

It has been said that the purpose to use probability masks was to aid the standard Matching Pursuit algorithm to code a determinate feature of the image with a certain type of atom. That is, to coordinate the use of these two dictionaries and, although, enhance the areas with more visual information. But we can also try to apply a single mask when coding an image with one dictionary. It has been shown that the use of probability masks never would lead to achieve better PSNR results but enhance the areas with more visual information (the main edges). The results shown in Figure 5.12 are those obtained when coding an image with a dictionary based on Anisotropic Refinement atoms using MP aided by a contour probability mask.

It can be said for all the images tested that the bit rate presented by the standard Matching Pursuit and the Hybrid Matching Pursuit decompositions are very close (± 0.005 bpp for various levels of quantization). This result is quite logical in the way that we use the same number of atoms, the same dictionaries and the coefficients follow approximately the same exponential decaying rule (see Figure 5.13). So, the statistics of the parameters do not change enough for the arithmetic encoder in order to take advantage of its statistical properties.

Finally, we have to add that the results here depicts the cases when the algorithm performs properly. As the results strongly depends on the masks (and they depends on the original image), in some cases, the codification does not give better results (both in terms of PSNR and visual quality).

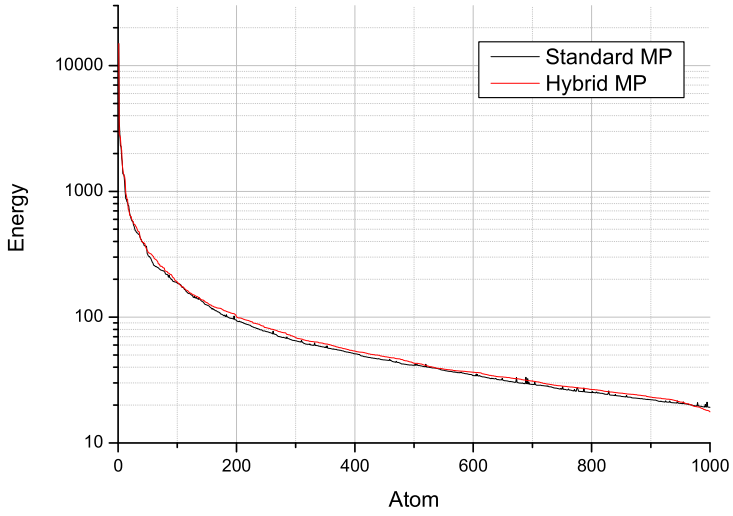
(a) (b) (c)

Figure 5.12: Hybrid Matching Pursuit with contour probability mask: results over LENA (128×128). From left to right: (a) Contour probability mask with $\alpha = 0.7$, (b) LENA coded with standard Matching Pursuit with 250 atoms and (c) LENA coded with hybrid Matching Pursuit. Results are showed in the table below.

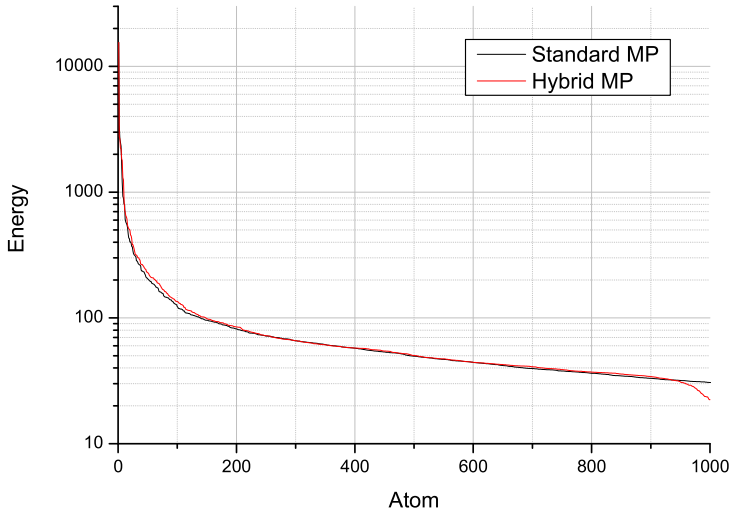
	PSNR	Bit rate for 256 levels	Poll for visual quality
MP	28.6151 dB	0.5512 bpp	16.66%
Hybrid MP	25.4818 dB	0.5503 bpp	83.33%

Multiple dictionary probability mask aided Matching Pursuit

The case of a Matching Pursuit decomposition over multiple dictionaries is a bit more complicated than the scheme presented in the former section. In this case, the use of probability masks is more justified recalling the fact that the Anisotropic Refinement atoms are more suitable to code edges and Gabor atoms to code textures. As these atoms have a very determinate function we have two options: to regard the placement of the atoms of each dictionary to the Matching Pursuit algorithm or aid it with the probability masks.



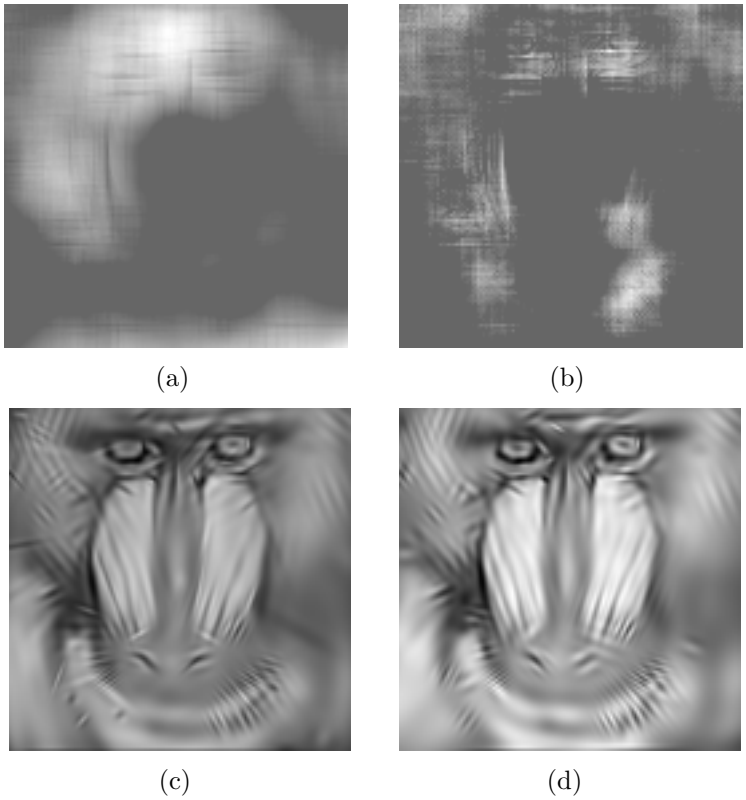
(a)



(b)

Figure 5.13: Energy of the coefficients in the standard Matching Pursuit and the Hybrid Matching Pursuit coders. In figure (a), coefficient's energy in the single dictionary decomposition and, in figure (b), multiple dictionary decomposition. Both follow a similar exponential decay that allows to apply the a posteriori exponential quantizer.

With this technique, situations like to try to code a texture by using Anisotropic Refinement atoms or code an edge by using Gabor atoms would be avoided. Also, the areas with more visual information (both contours and textures) will be enhanced.



(a) (b)

(c) (d)

Figure 5.14: Hybrid Matching Pursuit with contour and texture probability masks: results over BABOON (128×128). In figures (a) and (b), the contour mask and the texture mask with $\alpha = 0.6$ and $\gamma = 0.3$ respectively. Figure (c) and (d) depicts 200 atoms reconstruction for the standard Matching Pursuit and the Hybrid Matching Pursuit respectively.

	PSNR	Bit rate for 256 levels	Poll for visual quality
MP	25.95 dB	0.4942 bpp	25%
Hybrid MP	22.34 dB	0.4893 bpp	75%

Regarding with the codification matter, the same assumptions of the former results are applicable to this example. The exponential decay of the coefficients follows a similar rule to the standard Matching Pursuit, hence the results are extremely close.

Finally, we have to add again that the results vary among the different test images. For some, the hybrid algorithm performs better (always speaking in terms of visual quality) and in some, it does not (see Figure 5.15). We will comment this effect in the conclusions chapter.

(a) (b)

Figure 5.15: Hybrid Matching Pursuit coder, example of under-performance. Figure (a) depicts LENA coded with MP and 200 atoms (under the same conditions as Figure 5.14). Figure (b) the codification with 200 atoms with Hybrid-MP. Clearly, image (a) has better visual quality than (b).

5.4 Scrambled Matching Pursuit

5.4.1 Introduction

A new idea arose while developing this thesis: Matching Pursuit gives a good sparse approximation of an image but not always this approximation choose the atoms that are more suitable in terms of HVS perception. A solution to this problem would be to take the set of atoms given by the Matching Pursuit algorithm and reorder (scramble) them to achieve a reconstruction more adequate to our perception. To reorder the coefficients two criteria have been tested:

1. *MPQM maximization*: By choosing the atom that gives the best MPQM at each iteration. As discussed in previous sections, the MPQM maximization at each iteration does not lead to a global maximization of the MPQM metric. In this case, trying to reorder the atoms using this criteria gives a poor reconstruction (even in terms of visual quality), so this method was dismissed.

2. *Probability masks*: It has been shown that computing a Matching Pursuit decomposition aided by probability masks over a dictionary based on AR atoms gives interesting results in the way our image is decomposed taken into account the strongest edges (hence, where more visual information is located). In the same way, the coefficients that have been found by an standard Matching Pursuit decomposition might be reordered in order to empathize these strong edges. This procedure could be extended to deal with double dictionaries (i.e. Gabor atoms and Anisotropic Refinement atoms).

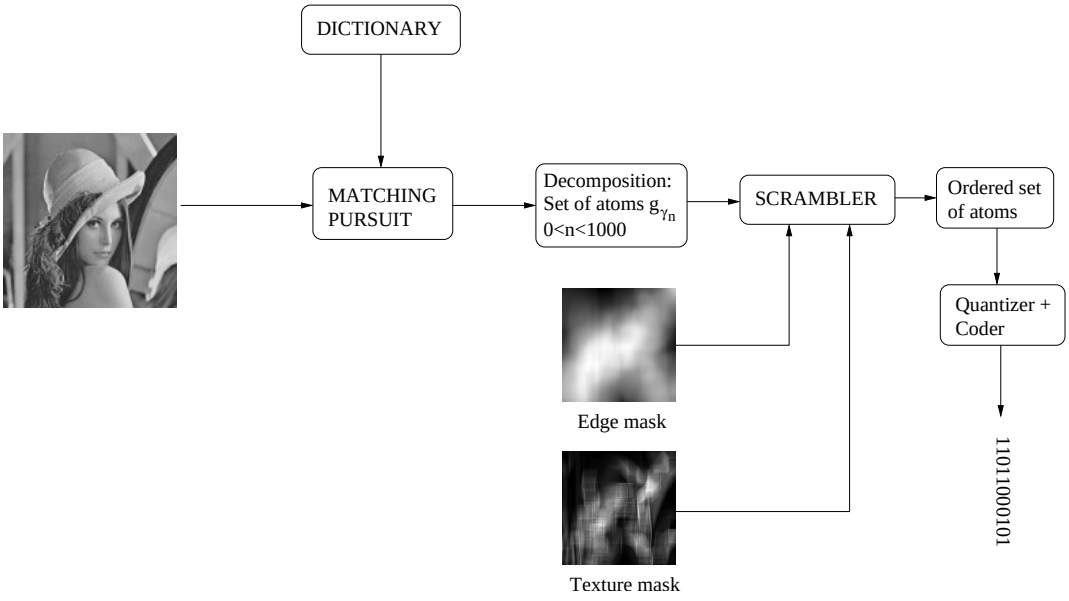


Figure 5.16: Scrambled Matching Pursuit coder.

5.4.2 Scrambled Matching Pursuit coder

Scrambled Matching Pursuit coder depicted in Figure 5.16 works as follows: first, the standard Matching Pursuit algorithm is computed and then the coefficients are reordered taking into account our probability masks that model and enhance the areas with more HVS information

One of the properties of Matching Pursuit is that gives an exponential decay of the coefficients and its quantification can be efficiently coded [37]. With our scrambling technique, this strictly exponential decay has been lost, as depicted in the Figure 5.17. Hence another quantization scheme must be found and [69] brings us a suitable tool to solve this matter: the death zone quantization. This quantizer basically works out as follows: first, the atom with smallest scalar product is chosen and its value is subtracted to the rest of the atoms. This is done in order to reduce the range and spend more bits to code the area of interest. Once this operation is performed, the resulting atoms are uniformly quantized.

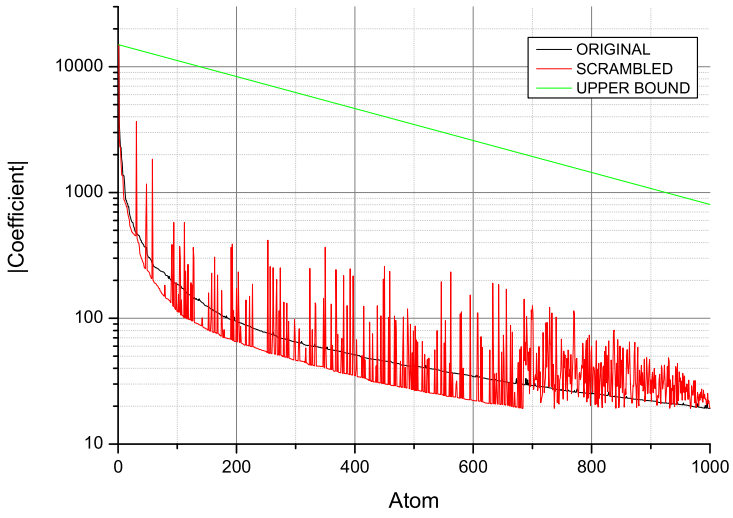


Figure 5.17: Scrambled Matching Pursuit coefficients' energy. The exponential decay of the coefficients achieved by the standard Matching Pursuit has been lost.

5.4.3 Results

The results regarding the effectiveness of this coder are showed in Figures 5.18, 5.19 and 5.20. The performance of this coder is based on the placement of atoms in the areas where more visual information is regarded and the results are positive. In fact, despite in all the cases, the PSNR ratio is lower than the one achieved by standard Matching Pursuit, the images reconstructed with Scrambled Matching Pursuit look more adequate for the

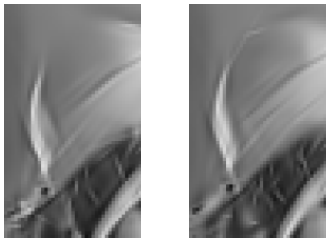
spectator. The results obtained over few test images show that in most of the cases people prefer the reconstruction built with this method.

If we focus on the results regarding the visual quality of the images, it would not be a fair comparison. Although the results are promising in terms of how "nice" look the images, we have to compare in terms of bit rate. Here is the main drawback: Scrambled Matching Pursuit does not take advantage of the optimized quantizer used by the standard Matching Pursuit. Nevertheless, for similar bit rates this new method achieves not so bad results, as showed in Figure 5.21.

(a) (b)

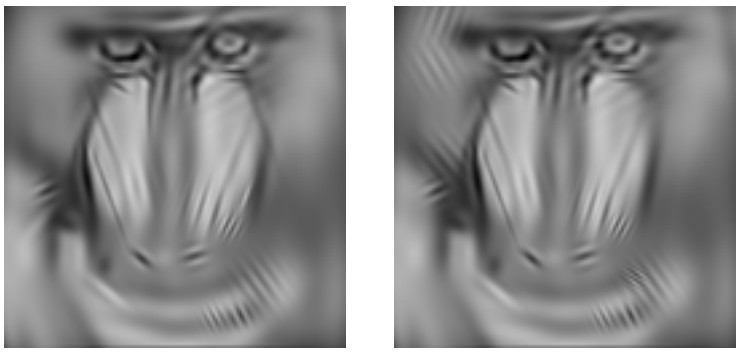
Figure 5.18: Scrambled Matching Pursuit results. Figure (a) is the reconstruction of LENA image with 450 atoms given by the standard flavor of Matching Pursuit and figure (b) is the Scrambled Matching Pursuit reconstruction with the same number of atoms. Results:

	PSNR	Poll for visual quality
MP	31.4 dB	33.3%
Scrambled MP	28.62 dB	66.6%



(a) (b)

Figure 5.19: Detail of Figure 5.18. The Scrambled version of Matching Pursuit leads to better visual results despite its lower PSNR ratio (compare the stripes of the hat and the texture of the fences).



(a) (b)

Figure 5.20: Scrambled Matching Pursuit results. Figure (a) is the reconstruction of BABOON with 100 atoms given by the standard Matching Pursuit and figure (b) is the Scrambled Matching Pursuit reconstruction with the same number of atoms. Results:

	PSNR	Poll for visual quality
MP	21.91 dB	8.3%
Scrambled MP	20.01 dB	91.6%

	Bitrate	Levels	Atoms	PSNR	PSNR (after Quantizing)
MP	0.9580 bpp	256	450	31.44 dB	29.01 dB
Scrambled MP (I)	0.9581 bpp	35	450	28.62 dB	21.18 dB
Scrambled MP (II)	0.9577 bpp	256	438	28.62 dB	27.05 dB

Figure 5.21: Comparative bit rate results for Scrambled Matching Pursuit. First row shows the results for the standard Matching Pursuit whereas second and third rows show the results for Scrambled Matching Pursuit with similar bit rates. Particularly, in the second row there are the results to achieve the same bit rate with the same number of atoms that leads to a have too less quantification levels. Finally, in the third row, the results with the same number of quantification levels but the number of atoms must decrease.

5.5 Split Matching Pursuit

5.5.1 Introduction

One of the main disadvantages of Matching Pursuit is the heavy computational load it demands to perform a decomposition. Moreover, another drawback is the fact that the dictionaries used to perform the decomposition are designed for the entire image. Imagine, a very hypothetic case, where our target image has the following characteristics: in a determinate area there are *only* edges and in an other area (not overlapped with the former) there are *only* textures. Thus, we could take advantage of this property and perform the Matching Pursuit decomposition just only with one of the dictionaries instead of both on each area.

With this idea, a new flavor of Matching Pursuit decomposition can be designed. Let I be a target image with $I_k|_{k=0}^{N-1}$ non-overlapped areas. Hence:

$$I = \bigcup_{k=0}^{N-1} I_k. \tag{5.20}$$

Then, if we assume that each area has just one determinate feature (edges or textures), a coarse approximation would be to perform a Matching Pursuit decomposition over each I_k with a *local dictionary*. This local dictionary would be, in a first approach, an AR or Gabor Dictionary. Thus, Matching Pursuit can be performed over each area separately and after merge all the areas together to generate a $\widetilde{I}$ approximation of the original image. This introductory approach to Split Matching Pursuit is very coarse in the sense that we do not pay attention to block artifacts introduced by the splitting process or any other effect. Despite this, if we just pay attention to the computation of the decomposition, this new algorithm reduces significantly the necessary operations. According to the Figure 5.22, we can calculate the number of operations (by using the FFT-MP optimization) to compute one atom for each method. Take in consideration a squared image of 128 pixels and a division in two identical halves for the Splitted Matching Pursuit:

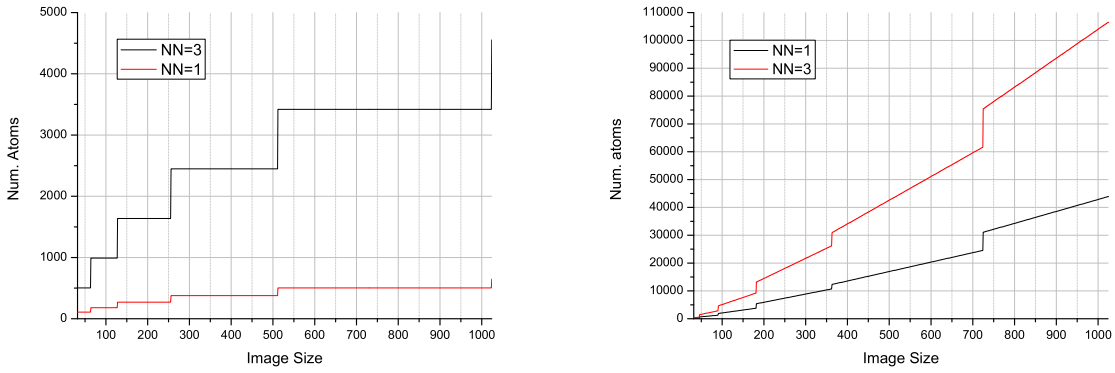
$$\text{Operations for MP} \quad \rightsquigarrow \quad \propto N_x^2 \cdot \mathcal{N}_{\text{AR}}(N_x \times N_x) = 53.673.984 \text{ operations}$$

$$\text{Operations for SMP} \quad \rightsquigarrow \quad \propto 2 \cdot \tfrac{N_x}{2} N_x \cdot \mathcal{N}_{\text{AR}}(N_x \times \tfrac{N_x}{2}) = 32.440.320 \text{ operations}$$

where $\mathcal{N}_{\text{AR}}(N_x, N_y)$ gives the size of an AR dictionary designed for an area of $N_x \times N_y$ pixels. According to this, the overall computation has been reduced.

Another advantage of this coding scheme is the possibility that, with the same bitrate used to code an image with the standard flavor of Matching Pursuit, we can achieve better performance. Relying on the fact that the local dictionaries are much more smaller that the original dictionary, strategies would be roughly defined as follows:

- To enlarge the local dictionaries to approximately reach the same number of functions of the dictionary used by standard Matching Pursuit. The criteria to make this



(a) (b)

Figure 5.22: Number of atoms for each dictionary in relation with the size of the image. In figure (a), $\mathcal{N}_{\text{AR}}$ dictionary size for a squared image of N_x pixels. In figure (b), $\mathcal{N}_{\text{Gabor}}$ dictionary size for a squared image of N_x pixels.

enlargement would be to add more orientations or scales, for instance. With this procedure, using approximately the same bitrate, we would catch more efficiently the structures of each partition.

- Instead of enlarging the local dictionaries we could place more atoms per region, with the same bitrate. This strategy also leads to better performance of the algorithm.

5.5.2 Split Matching Pursuit coder

Let us formalize the procedure to perform the Splitted Matching Pursuit according with the description of Figure 5.23:

Partition The first step is to define a set of N non-overlapped areas from the original image with homogeneous properties into them. The homogeneity criteria would be: whether there is a high density of edges or a high density of textures or patterns or there are both. The correct election of a segmentation algorithm that can define a partition of the image with such criteria will be the key to achieve a good performance of the decomposition. Various segmentation schemes have been tested during the pursuing of this work but none has showed a good performance. Instead of that, a by-hand segmentation has been done. See Future Work section.

Transition area In order to avoid the artifacts introduced by the splitting of the original image, a transition area is defined. This area will enlarge the borders of the partitions, overlapping between them. Let us call ξ the transition width around a partition. Once this factor is set, the enlarged partitions $I_k^{\xi}|_{k=0}^{N-1}$ will be our operational units. So,

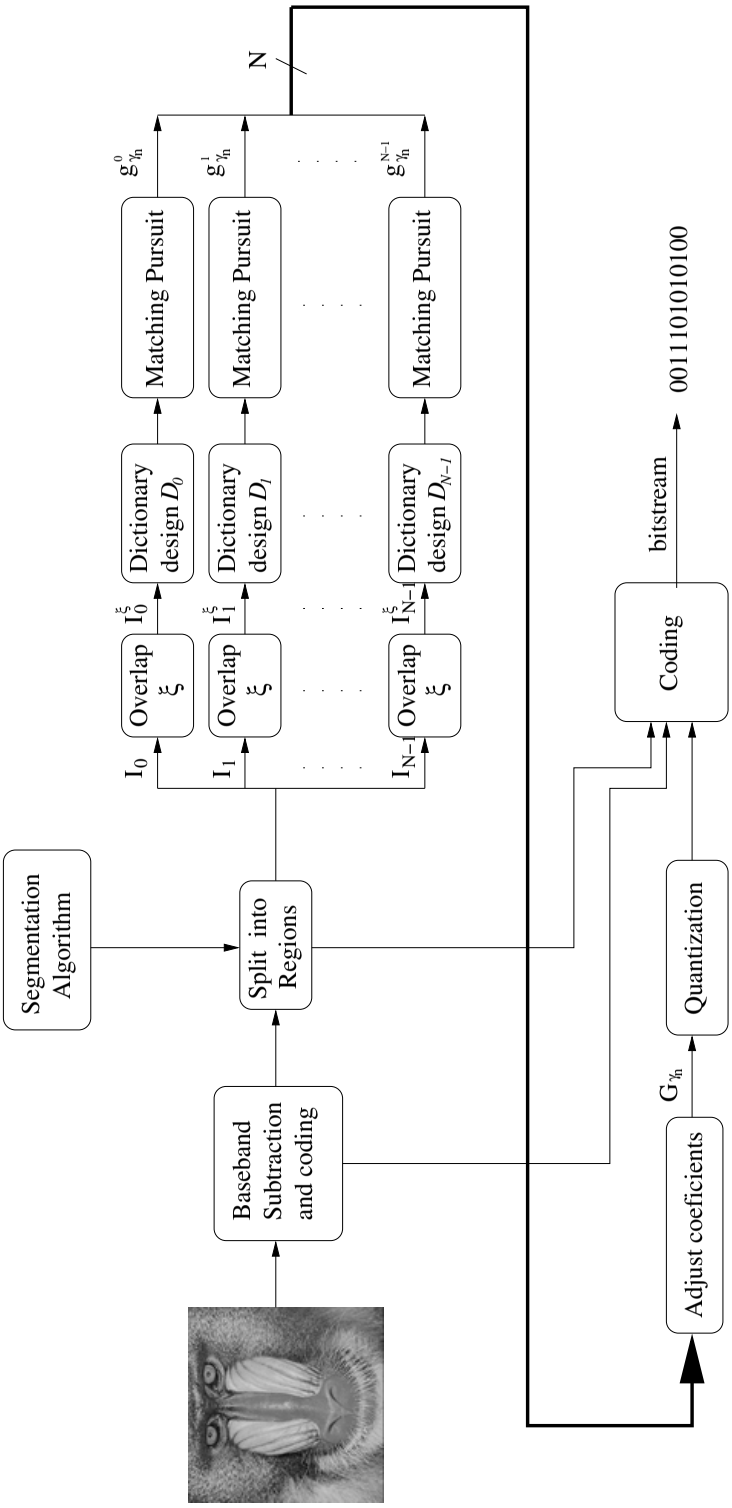


Figure 5.23: Splitted Matching Pursuit coder.

our Matching Pursuit decomposition will be performed over the enlarged partitions instead of the original ones. Finally, this transition area is added in order to smooth the junctions between adjacent partitions when the reconstruction takes place.

Dictionary definition Once we have set the partitions of the image, we can extract information about them in order to define a local dictionary $\mathcal{D}_k$ as matched as possible with the structures present in the image. Here there would be many methods to define such dictionaries but, to make a simple analysis, let us define these dictionaries in the same way we defined the whole dictionary. The dictionaries can be enlarged by adding more orientations or scales, depending on the bitrate we would like to have. At this point, we have to recall that our dictionaries are defined over rectangular areas, so the partitions must be defined accordingly. The improvement achieved is that the local dictionaries size is drastically reduced, hence the speed-up of the algorithm.

Matching Pursuit Matching Pursuit decomposition is performed over the N partitions with their respective local dictionaries $\mathcal{D}_k$. When performing the Matching Pursuit, we have to decide how many atoms will be computed per each area k. If we want to compute M atoms for the whole image, an equitable method would be to divide them in function of the pixels each area has:

$$\text{Atoms}_k = \frac{M}{N_x N_y} \cdot \text{Area}(I_k) + \text{Extra atoms (bitrate)}, \tag{5.21}$$

where N_x and N_y are the dimensions of the original image. The extra atoms noted above refer to the atoms we can add to reach the same bitrate achieved by the standard Matching Pursuit. Note that we compute the number of atoms of each area in function of I_k instead of I_k^ξ. For future manipulations, let us call $g_{\gamma_n}^k$ the set of atoms obtained after the Matching Pursuit decomposition for the partition k.

Coefficients adjusting When the set of coefficients $g_{\gamma_n}^k$ for all the partitions have been obtained we have to create a global description of all the sets. That is to merge the k sets of coefficients to build an unique set G_{γ_n}. Each partition, when performing Matching Pursuit, have its own coordinates reference, hence the global reference with the original image coordinates origin has been lost. To adjust all the k sets to unify the origin, we have just to add the convenient offsets to the positions of the atoms of each section in reference with the original image. It has been tested that by coding all the coefficients together is more efficient that coding them separately.

Quantification Once G_{γ_n} is built we perform the quantification. It has been already shown in the former section that Scrambled Matching Pursuit had the drawback that the coefficients do not follow the exponential decay, hence the efficient exponential a posteriori quantization can not be applied. The question that arises is: does Split Matching Pursuit follow the exponential decay, once the atom's coefficients have been gathered in G_{γ_n}? As depicted in Figure 5.24, they do. Hence the quantification can be performed efficiently as done in the standard Matching Pursuit coder.

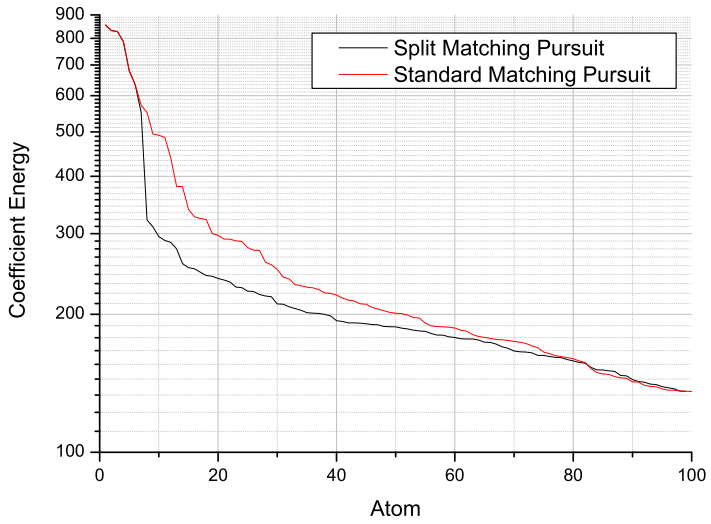


Figure 5.24: Coefficient's energy decay for Split Matching Pursuit. As showed below the energy follows a quasi-exponential rule. Hence the a posteriori quantification perform properly.

Codification The stream that will be sent must contain the quantized coefficients, the parameters of the atoms γ but, in this particular scheme, a description about the partitions (necessary to perform the reconstruction), the overlapping area ξ and the baseband as well.

Reconstruction The reconstruction of a Split Matching Pursuit algorithm is not direct as happened with standard Matching Pursuit. This is because the partitions are overlapped and in the areas where the partitions overlap there would be more energy. Hence, a way to perform the reconstruction would be as:

$$\widetilde{I} = \bigcup_{k=0}^{N-1} \widetilde{I}_k^{\xi} - \bigcap_{k=0}^{N-1} \bigcap_{l=0}^{N-1} \left(\underbrace{\widetilde{I}_k^{\xi}, \widetilde{I}_l^{\xi}}_{k \neq l} \right) + \Upsilon \left(\bigcap_{k=0}^{N-1} \bigcap_{l=0}^{N-1} \left(\underbrace{\widetilde{I}_k^{\xi}, \widetilde{I}_l^{\xi}}_{k \neq l} \right) \right), \tag{5.22}$$

where $\Upsilon(\cdot) \in \mathcal{C}^1$ is a smoothing function. In fact, the utility of this equation is to smooth the overlapped areas in order to avoid the block artifacts and restore the energy. In our particular case, a first-order polynomial function has been chosen.

5.5.3 Results

The results concerning the performance of Split Matching Pursuit are shown in two figures: 5.25 and 5.26. These figures, will illustrate the behavior and performance of the scheme.

From Figure 5.25 we can comment some remarkable facts:

- When applying standard Matching Pursuit over regions with very differentiate featured regions, the visual quality is far from the best (see figure (c)). 100 atoms have been placed in all the image.

- In subfigure (d), Matching Pursuit has been applied independently over each region and after they have been merged (50 atoms at each region). Visually, the results improve significantly (see how better the fingerprint been represented has). But the election to place two dictionaries lead to ringing effects into the contour area (due to the Gabor atoms). Hence, the suitability to use local dictionaries.

- In subfigure (e), a Gabor dictionary has been used to code the fingerprint and an AR dictionary to code the lines. The ringing effects in the lines have disappeared and the total visual quality has improved significantly.

About this example, some data can be computed:

Algorithm	PSNR	Bit rate Levels=64	Bit rate Levels=128	Bit rate Levels=512	Time Consumed
Standard MP	20.86 dB	0.2227 bpp	0.229 bpp	0.2529 bpp	35 mins.
Split MP (2 dictionaries)	20.43 dB	0.2251 bpp	0.2344 bpp	0.2573 bpp	15 mins.
Split MP (local dictionaries)	20.26 dB	0.2173 bpp	0.226 bpp	0.2485 bpp	6 mins.

As we can see, the comparison leads to the following points:

- The PSNR ratio is very close in all cases. For more test images it has been shown that it behaves alike this in most of the cases (taking into account that we will have better results as better is our segmentation).

- The time demanded, to achieve these close PSNR ratios, is quite different. With Split Matching Pursuit driven by local dictionaries the time has been reduced almost 80% in comparison with the Standard Matching Pursuit.

- The bit rate achieved by Split Matching Pursuit with local dictionaries is always smaller than the bit rate demanded by Standard Matching Pursuit (even taking into account that the header of Split Matching Pursuit is bigger due to the code of the partition's description). The reason is because the smaller dictionaries we use the smaller will be the coding range. This leads to smaller bit rates for our Split Matching Pursuit decompositions.

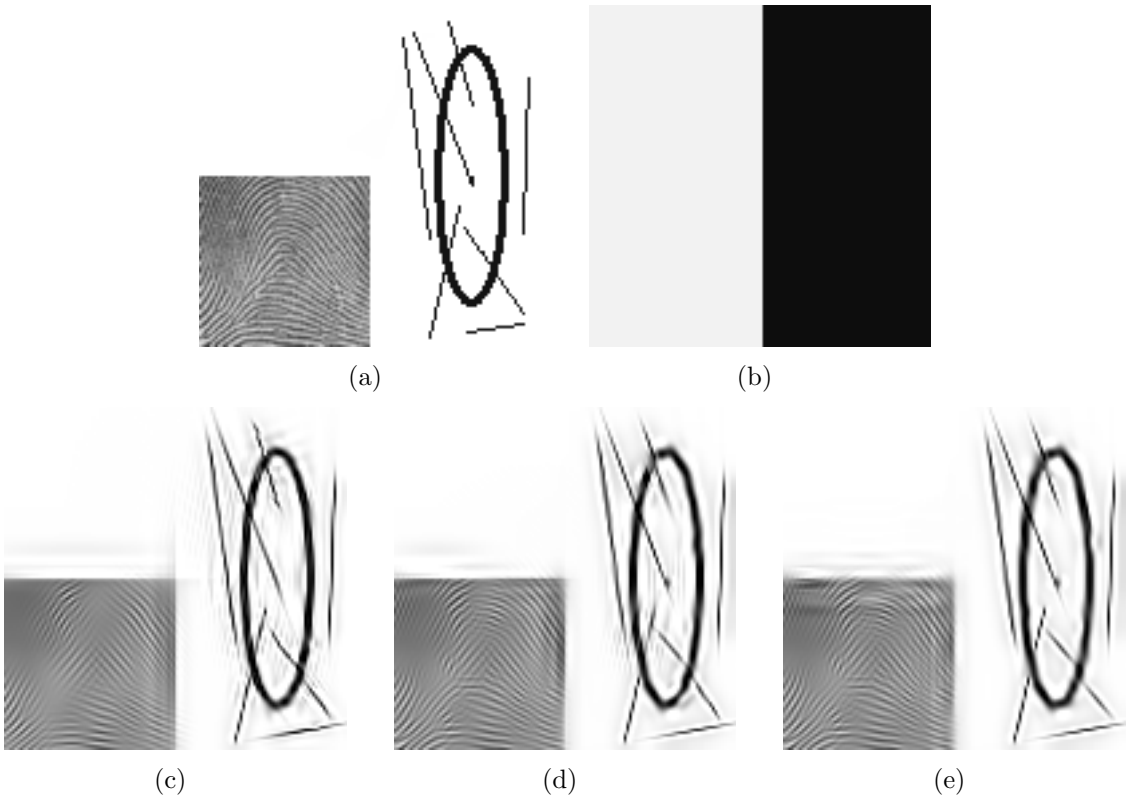


Figure 5.25: First example of Split Matching Pursuit. In figures (a) and (b) are depicted the original image and its splitting mask respectively. Note that the image has two differentiate areas: the left with the fingerpring (textures) and the right with the lines (contours). Figure (c) is the standard Matching Pursuit (applied over all the image), figure (d) the Split Matching Pursuit is applied with both dictionaries over each partition and figure (e) is the Split Matching Pursuit applied with local dictionaries over each region. Every image has placed 100 atoms.

In figure 5.26 there is shown the performance of Split Matching Pursuit for a more natural image. The results obtained for the decompositions were similar to the obtained for the previous example:

Algorithm	PSNR	Bit rate (R) Levels=256	Time Consumed
Standard MP	25.01 dB	0.2515 bpp	35 mins.
Split MP (local dictionaries)	23.97 dB	0.2380 bpp	7 mins.

With this figure, we give an example of how to add more atoms to the Split Matching Pursuit decomposition to achieve the same bit rate as standard Matching Pursuit. It has been shown that Split Matching Pursuit leads to lower bit rates than the standard method, hence, we can calculate this excess as:

$$\Delta R = 0.2515 - 0.2380 = 0.0135 \text{ bpp} \tag{5.23}$$

Then, to find how atoms we can add more to the Split decomposition, we would calculate the bit rate per pixel:

$$\frac{R_{\text{Split}}}{\text{Num.atoms}} = \frac{0.238}{100} = 0,00238 \text{ bpp/atom} \tag{5.24}$$

Finally, we can calculate the extra atoms by:

$$\frac{\Delta R}{0,00238} \approx 6 \tag{5.25}$$

By adding these 6 more atoms, the Split Matching Pursuit decomposition (with the same bit rate as the Matching Pursuit one) achieves a PSNR of 24.13 dB.

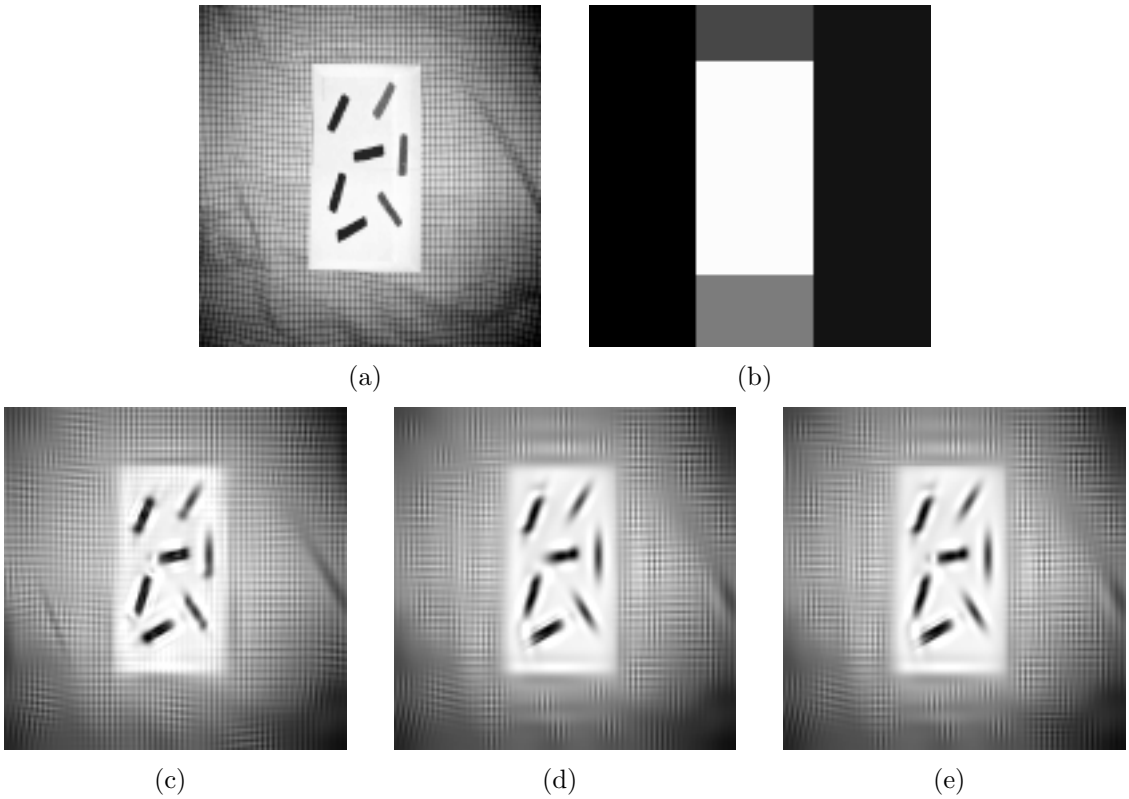


Figure 5.26: Second example of Split Matching Pursuit. In figures (a) and (b) are depicted the original image and its splitting mask respectively. In figure (c), Matching Pursuit decomposition with 100 atoms; figure (d), Split Matching Pursuit decomposition with 100 atoms; figure (e), Split Matching Pursuit decomposition with 106 atoms

Chapter 6

Conclusions and Future work

6.1 Achievements

We did already know that Matching Pursuit was an efficient approach to very low bit rate coding of images. But the original algorithm does not take into account the visual information contained in the images. Hence, the algorithm has been modified to include this issue into the coding process by introducing two probability masks that will "help" the standard Matching Pursuit algorithm to place more atoms in the areas of interest (where there are strong edges and textures). With this idea, three new coding schemes have been proposed:

Hybrid Matching Pursuit that ponders the scalar products of the atoms in order to enhance the areas with more visual information.

Scrambled Matching Pursuit that takes the original stream given by a Matching Pursuit decomposition and scrambles it in order to place the atoms that regard more visual information.

Split Matching Pursuit is another version of Matching Pursuit that performs the decomposition in blocks. The advantage of this scheme is that achieves a similar performance (in PSNR terms) than the standard Matching Pursuit but takes around 60-80% time less.

6.1.1 Conclusions

Matching Pursuit vs Hybrid Matching Pursuit

It can be demonstrated that we will never achieve better PSNR results with this Hybrid Matching Pursuit than with the standard Matching Pursuit. But in some cases, the decompositions given by this new scheme leads to representations that look nicer for the observer due to the enhancement of the areas with most of visual information. But it does not happen always. The main problem is that this problem is ill-posed. Due to the heavy

dependence on the definition of the probability masks with the original image, the results may vary from one case to another. Furthermore, there is the problem of adjusting a set of parameters (the mutual mask influence γ, the waterfilling parameter α,...) that controls the behavior of the algorithm. There is no analytic method to set this parameters so, for each image, the optimal values are different. Hence, if there would be a reliable method to find the optimal parameters, this scheme would perform visually better decompositions than standard Matching Pursuit.

Moreover, we have to add that the results already given by the standard Matching Pursuit are also good in terms of visual quality. That means that despite it does not take into account the more meaningful areas of the image, the decompositions are visually nice. The reason relies into the fact that the dictionary used (the one based on Anisotropic Refinement atoms) has a similar response than the early cortex visual cells, hence there exists a high correlation between the decomposition and how the images are perceived by our eyes.

A few words may be added about scalability. Matching Pursuit has been found as a good choice in terms of scalability [38]. This new scheme also shares this property in the sense that they have a similar statistical behavior (that would affect the quantization) and properties.

Matching Pursuit vs Scrambled Matching Pursuit

This new scheme seems to be a good choice when we want to design a bit stream that leads to reconstructions that increase the visual quality proportionally to the number of atoms. The idea of scrambling the atoms (controlled by the probability masks) leads to a reconstruction that achieves better visual quality proportionally to the number of atoms and has the advantage of not introducing an extra heavy load into the coding process. The results seem promising because there is a significant improvement of the visual quality in relation with the standard decompositions. It would be interesting when designing image receptors (for example in a cell phone) to have a reconstruction in such a way.

The disadvantage of this scheme is the lost of the exponential decay of the coefficients when reordering them. Hence, the efficient exponential adaptive quantization can not be performed and the bit rate increases a bit. However, quantization with death-zone technique give also good results and this Scrambled Matching Pursuit becomes a reliable strategy when coding image at very low bit-rate focusing on a "visually scaled" bitstream.

Matching Pursuit vs Split Matching Pursuit

When performing Matching Pursuit decompositions, the heavy computational load leads to wait a long time. Hence, improvements to reduce the computational load must be done. This technique by splitting the image in sub-images[1] and performing the Matching Pursuit decomposition over each sub-image, the computing time is reduced around 60-80%. Then,

[1]This technique has been already used in video coding with Matching Pursuit by [3]

this technique can be a solution to implement reliable Matching Pursuit coders in a near future.

Other advantages can be cited. The decay of the coefficients continues followinging an exponential rule, hence the efficient a posteriori quantization can be done. Furthermore, as the area where Matching Pursuit is computed has been reduced, the dictionaries are also reduced. This implies that our coding range is smaller, hence there is a reduction on the overall bit rate.

In terms of visual quality, there is an improvement. Once we have split the original image, the sub-images are coded with dictionaries created in the same way they are created for the whole image. These local dictionaries are smaller than the global ones, hence we can enlarge them by adding scales or rotations to catch better the structures presents in the sub-image.

Two drawbacks have been found in this new coder. The first is the splitting process: there must be an efficient segmentation algorithm to split the image and decide which dictionaries must be place at each segment. That is not easy and we could not find a good method to perform this segmentation. Hence, there is still a hole to fill... The other drawback is the increase of the complexity of the receptor. When the receptor decodes an image coded with this scheme, it must built the partitions in order to apply the smoothing functions at the edges. This implies a bit more of complexity than the decoder for Matching Pursuit coded images.

6.2 Future work

Sparse approximation of functions is a science field that has been studied thoroughly during the last years but there is still a lot of unexplored possibilities.

- Many parameters are involved into the definition of these Matching Pursuit proposed in this thesis: the mutual mask influence γ, the waterfilling parameter α,... It would be interesting to find a reliable method to assign values to these parameters.

- Study more thoroughly the behavior of the Babel function. Concretely, study for different types of dictionaries the behavior of

$$\frac{\partial \mu_1(m)}{\partial m}. \tag{6.1}$$

  Also, would be interesting to check if there is any concrete value for

$$\lim_{m\to\infty} \frac{\partial \mu_1(m)}{\partial m}, \tag{6.2}$$

  and if it has any relevance. With this knowledge we could try to design dictionaries that would fulfill some properties with the Babel function.

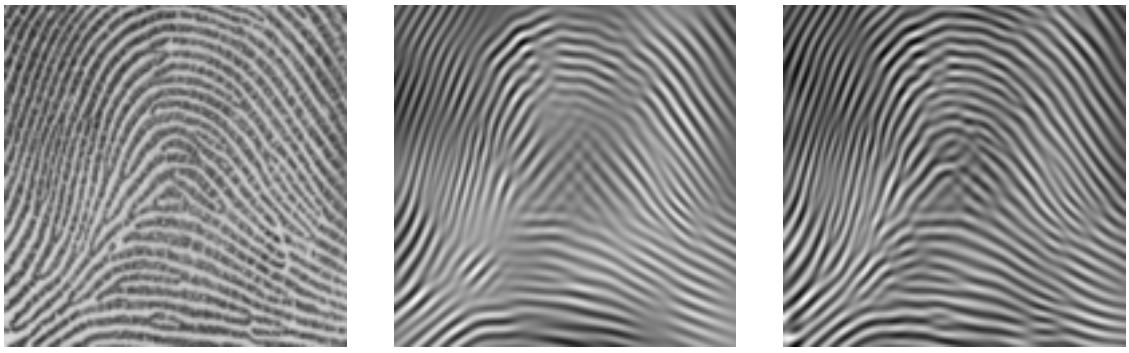
- It could be interesting to find other methods to define the probability masks. In this thesis, the edge probability mask $\mathcal{P}_E(x, y)$ was based on the combination of an edge detector (based on contrast measures) and a model describing the behavior of Matching Pursuit to shape an edge. In the other hand, the texture probability mask $\mathcal{P}_T(x, y)$ was based on Simoncelli's texture detector based on co-occurrence matrix features. Either edge or texture probability masks could be initially defined by using other methods. For example, there is abundant literature [65] describing various methods for texture analysis: there could be methods that would give better texture segmentation. Furthermore, an iterative method to create the probability masks could be defined.

- It has been shown in this thesis that the use of a dictionary based on Gabor functions designed to code the textures present in an image leads to good results (even in PSNR and HVS metrics). As we could appreciate in Chapter 4, all the dictionaries used all over our MP decompositions ($\mathcal{D}_G$,$\mathcal{D}_{AR}$ and $\mathcal{D}_{Ga}$) show a high coherence μ parameter (due to its high redundancy). Anisotropic Refinement atoms are appropriate to code the edges of the image and Gauss atoms to code image baseband but more efficient dictionaries to code the texture could be tested. In fact, the Gabor dictionary is suitable to code textures but, in some cases, it takes a large amount of atoms to code a simple pattern. Other dictionaries with functions more affine to the textures that can be found in natural images could be designed. Some alternatives to be tested are: dictionaries based on Markov Random Fields, parametric texture functions or even a taylor-made dictionary of textures (with dilation and rotation properties).

- The use of Gabor atoms to code efficiently structures has been largely proved in this dissertation. A determinate type of images that are very textured are fingerprints. Hence, Gabor atoms would be suitable to code fingerprints achieving high compression ratios. Even identification methods based on the good properties of Matching Pursuit decomposition such as dilation and orientation invariance could be performed. See Figure 6.1.

- As the reader has read in Section 3.1, a function can be decomposed into a sum of terms but, to obtain the optimal approximation (these that achieves a minimum quadratic error), it leads to a NP-hard problem. This is the reason why we choose to apply a more tractable procedure: the Matching Pursuit suboptimal approximation. In order to speed-up this algorithm there would be few issues than can be improved:

  - Take more than one atom each iteration. In fact, when this section was being written, this method was being tested in [75], [60].
  - We would take advantage of the probability masks (in the supposition that they have defined properly). Atoms belonging to different dictionaries could be placed in the same iteration if the masks show that there is a spatial exclusion between them.



(a) (b) (c)

Figure 6.1: Fingerprints coding. Subfigure (a), original fingerprint; figure (b), fingerprint coded with 50 Gabor atoms; figure (c), fingerprint coded with 100 atoms. It is showed how with very few atoms, an efficient coding of fingerprints can be done with Matching Pursuit.

- This thesis showed that the use of probability masks to enhance some areas with a noticeable visual quality lead to *nicer* representations of the image (in terms of visual quality). In the same way, we would design a Matching Pursuit coder to code a determinate type of images, for example faces. Then, instead of a probability mask for edges and textures, we would use masks to enhance the areas were there are placed the eyes, nose, mouth,... The idea to focus the coding in the places that have more *information* mimics what MPEG4 coders do, that gives more bits to the objects of interest.

- Into the Split Matching Pursuit coder, we showed that this method achieves good performance in the sense that takes much more less time to compute a decomposition with similar PSNR ratio. Few improvement would be done:

  - Find out what would be the best segmentation method for this coder.
  - Perform the codification by layers. That is, to divide the image in few layers, each one with one determinate feature and code them separately. After merge all them (taking in account the energy conservation) to have the reconstructed image.

Appendix A

Exponential convergence to 0 of the residual in MP

Let be $R^n f$ the approximation error of f after choosing n vectors in the dictionary and the energy of this error is given by

$$\|R^n f\|^2 = \|f\|^2 - \sum_{k=0}^{n-1} \left|\langle R^k f, g_{\gamma_k}\rangle\right|^2 . \tag{A.1}$$

Let be $\mathcal{H}$ a Hilbert Space, then, for any $f \in \mathcal{H}$, the convergente of the error to zero is shown in [96] to be a consequence of a theorem proved by [53]. Here is a detailed demostration of the exponential convergence to 0 of the residual in MP.

Theorem A.0.1 *There exists $\lambda > 0$ such that for all $m \geq 0$and $\forall f \in \mathbb{C}^N$:*

$$\|R^m f\| \leq 2^{-\lambda m}\|f\|. \tag{A.2}$$

As a consequence

$$f = \sum_{m=0}^{+\infty} \langle R^m f, g_{\gamma_m}\rangle g_{\gamma_m}, \tag{A.3}$$

and

$$\|f\|^2 = \sum_{m=0}^{+\infty} |\langle R^m f, g_{\gamma_m}\rangle|^2 , \tag{A.4}$$

where the convergence of 2 is intended in the strong sense.

Proof Let us verify that exists $\beta > 0$ such that for any $f \in \mathbb{C}^N$

$$\sup_{\gamma\in\Gamma} |\langle f_m, g_\gamma\rangle| \geq \beta\|f\|. \tag{A.5}$$

Supose that it is not possible to find such a β. This means that we can construct $\{f_m\}_{m\in\mathbb{N}}$ with $\|f_m\| = 1$ and

$$\lim_{m\to\infty} \sup_{\gamma\in\Gamma} |\langle f_m, g_\gamma\rangle| = 0. \tag{A.6}$$

Since the the unit sphere $\mathbb{C}^N$ is compact, there exists a subspace $\{f_{m_k}\}_{k\in\mathbb{N}}$ that converges to a unit vector $f \in \mathbb{C}^N$. It follows that

$$\sup_{\gamma\in\Gamma} |\langle f, g_\gamma\rangle| = 0, \tag{A.7}$$

so $\langle f, g_\gamma\rangle = 0$ for all $g_\gamma \in \mathcal{D}$. Since $\mathcal{D}$ contains a basis of $\mathbb{C}^N$, necessarily $f = 0$ which is not possible because $\|f\| = 1$. This proves that our initial assumption is wrong and, hence, there exists β such that A.5 holds.

The decay condition A.2 is derived from the energy conservation:

$$\|R^{m+1}f\|^2 = \|R^m f\|^2 - |\langle R^m f, g_{\gamma_m}\rangle|\,. \tag{A.8}$$

The Matching Pursuit chooses g_{γ_m} that satisfies

$$\langle R^m f, g_{\gamma_m}\rangle \geq \alpha \sup_{\gamma\in\Gamma} |\langle R^m f, g_\gamma\rangle|\,, \tag{A.9}$$

and A.5 implies that $\langle R^m f, g_{\gamma_m}\rangle \geq \alpha\beta\|R^m f\|$. So

$$\|R^{m+1}\| \leq \|R^m f\|\sqrt{1-\alpha^2\beta^2}, \tag{A.10}$$

which verifies A.2 for

$$2^{-\lambda} = \sqrt{1-\alpha^2\beta^2} < 1. \tag{A.11}$$

This also proves that

$$\lim_{m\to\infty} \|R^m f\| = 0. \tag{A.12}$$

Appendix B

Babel function for a Gaussian Dictionary $\mathcal{D}_G$

To calculate the Babel function of a generic atom, we have to apply the Babel function definition introduced by [88]

$$\mu_1(m) = \max_{|\Lambda|=m} \max_{\psi} \sum_{\Lambda} |\langle \psi, g_\gamma \rangle| , \tag{B.1}$$

where the vector ψ ranges over the atoms indexed by the complementary of Λ (Ω/Λ). We recall the simplified expresion of this equation as

$$\mu_1(m) = \max_{|\Lambda|=m} \max_{k \notin \Lambda} \sum_{j \in \Lambda} \left| \langle g_{\gamma_k}, g_{\gamma_j} \rangle \right| . \tag{B.2}$$

Then, to make a feasible analytical calculation of this function we must take in consideration some constrains that would allow us to find how to calculate a lower bound of it. In the particular case of the Gaussian Dictionary $\mathcal{D}_G$, there are two parameters that define the shape of the atom $\gamma = (\boldsymbol{p}, \sigma)$. First we would choose an atom g_{γ_0} that would be most probable to achieve the maximum sum of scalar products $\sum_{j \in \Lambda} \left| \langle g_{\gamma_k}, g_{\gamma_j} \rangle \right|$. The most suitable atom for this choice is the atom g_{γ_0} with position $\boldsymbol{p} = (\frac{N_x}{2}, \frac{N_y}{2})$, where $N_x \times N_y$ is the size of the image, and the maximum σ allowed by the scales.

Once we have placed the reference atom g_{γ_0}, we will have to define a sequence for the chosen $g_{\gamma_j}|_{j=1}^m$ that would maximize the sumatory. The first constraint to define this sequence arises easily: to place the $g_{\gamma_j}|_{j=1}^m$ atoms with the smallest distances from g_{γ_0}. In this way, the g_{γ_j}'s atoms will have more overlapping area and, hence, bigger scalar product. So, that is

$$\min \|(x_j, y_j) - (x_0, y_0)\|_{j=1}^m. \tag{B.3}$$

The constraint over the parameter σ would be as follows: to take the σ's in decreasing order in each point selected. Hence, we have that the election order for g_{γ_j}'s atoms the will be by choosing the atoms belonging to concentric coronas around the central pixel

with decreasing σ's at each pixel. Finally, to exemplify this process, we would write the following pseudo-code:

```
centerx:=Nx/2;
centery:=Ny/2;
m:=0;

for radius:=0 to min(Nx/2,Ny/2)
{
    for kx:=-radius to radius
    {
        for ky:=-radius to radius
        {
            if abs(kx)!=radi or abs(ky)!=radi
                break;
            for sigma:=sigma_min to sigma_max
            {
                atom(x,m):=centerx+kx;
                atom(y,m):=centery+ky;
                atom(sigma,m):=sigma;
                m++;
            }
        }
    }
}
```

A symbolic assignment of the atoms for $m = 17$ would be as follows:

$(4)(N_x-1, N_y-1, \sigma_0)$	$(6)(N_x, N_y-1, \sigma_0)$	$(8)(N_x+1, N_y-1, \sigma_0)$
$(5)(N_x-1, N_y-1, \sigma_1)$	$(7)(N_x, N_y-1, \sigma_1)$	$(9)(N_x+1, N_y-1, \sigma_1)$
$(2)(N_x-1, N_y, \sigma_0)$	$(0)\ (N_x, N_y, \sigma_0)$	$(10)(N_x+1, N_y, \sigma_0)$
$(3)(N_x-1, N_y, \sigma_1)$	$(1)\ (N_x, N_y, \sigma_1)$	$(11)(N_x+1, N_y, \sigma_1)$
$(16)(N_x-1, N_y+1, \sigma_0)$	$(14)(N_x, N_y+1, \sigma_0)$	$(12)(N_x+1, N_y+1, \sigma_0)$
$(17)(N_x-1, N_y+1, \sigma_1)$	$(15)(N_x, N_y+1, \sigma_1)$	$(13)(N_x+1, N_y+1, \sigma_1)$

where $\{\sigma_0, \sigma_1\}$ are the σ's parameters in decreasing order and the first number within brackets the order the atoms have been chosen.

Appendix C

Properties of Matching Pursuit

Translation invariance

A dictionary $\mathcal{D}$ is translation invariant if for any $g_\gamma[\vec{n}] \in \mathcal{D}$ and any $\vec{p} = [p_x, p_y] \in [0..N_x - 1, 0..N_y - 1]$ then $g_\gamma[\vec{n} - \vec{p}] \in \mathcal{D}$. If we compute the Matching Pursuit in a translation invariant dictionary, then it will be translation invariant. Given the matching of f in $\mathcal{D}$,

$$f[\vec{n}] = \sum_{m=0}^{M-1} \langle R^m f, g_{\gamma_m} \rangle g_{\gamma_m}[\vec{n}] + R^M f[\vec{n}], \tag{C.1}$$

it is easy to verify [21] that the Matching Pursuit of $f_{\vec{p}}[\vec{n}] = f[\vec{n} - \vec{p}]$ selects a translation by $\vec{p}$ of the same vectors g_{γ_m} with the same decomposition coefficients

$$f_{\vec{p}}[\vec{n}] = \sum_{m=0}^{M-1} \langle R^m f, g_{\gamma_m} \rangle g_{\gamma_m}[\vec{n} - \vec{p}] + R^M f_{\vec{p}}. \tag{C.2}$$

Rotation invariance

By analogy, we will obtain a rotation invariant Matching Pursuit when we use a rotation invariant dictionary $\mathcal{D}$. A dictionary is rotation invariant if for any $g_\gamma[\vec{n}] \in \mathcal{D}$ and any $\theta \in [0, 2\pi)$ then $g_\gamma[r_\theta \vec{n}] \in \mathcal{D}$ where r_θ is the rotation operator given by the matrix:

$$\begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}. \tag{C.3}$$

Given the decomposition of f in $\mathcal{D}$,

$$f[\vec{n}] = \sum_{m=0}^{M-1} \langle R^m f, g_{\gamma_m} \rangle g_{\gamma_m}[\vec{n}] + R^M f[\vec{n}], \tag{C.4}$$

one can verify that the Matching Pursuit of $f_\theta[\vec{n}] = f[r_\theta \cdot \vec{n}]$ selects a rotation by θ of the same vectors g_{γ_m} with the same decomposition coefficients:

$$f_\theta[\vec{n}] = \sum_{m=0}^{M-1} \langle R^m f, g_{\gamma_m} \rangle g_{\gamma_m}[r_\theta \cdot \vec{n}] + R^M f_\theta[\vec{n}] \tag{C.5}$$

This makes Matching Pursuit a useful technique to rotate images, because the only extra calculation we have to do is to modify the index of the reconstruction atoms when computing the coded image, instead of applying the rotation matrix to every pixel of the image.

Dilation invariance

As in the previous two cases, Matching Pursuit is dilation invariant if the dictionary of functions used by the pursuit is dilation invariant. A dictionary $\mathcal{D}$ is dilation invariant when for any $g_\gamma[\vec{n}] \in \mathcal{D}$ and any $s \in [0, s_{\max}]$ then $g_\gamma[\frac{\vec{n}}{s}] \in \mathcal{D}$. In this case, the matching pursuit of $f_s[\vec{n}] = f[\frac{\vec{n}}{s}]$ will select a dilation by s of the same vectors g_{γ_m} with the same decomposition coefficients:

$$f_s[\vec{n}] = \sum_{m=0}^{M-1} \langle R^m f, g_{\gamma_m} \rangle g_{\gamma_m} \left[\frac{\vec{n}}{s}\right] + R^M f_s[\vec{n}]. \tag{C.6}$$

One thing to take in account is that MP is dilation invariant only when the dilation is applied to the whole $\vec{n}$. If one applied a different scaling for every component of the vector, the property of dilation invariance will be lost.

Dilation invariance gives an easy way of scaling an image: the only thing that has to be done is to modify the scaling parameter (the same for x and y) when reconstructing the image and we will have a larger or smaller image.

Also if we join rotation, translation and dilation invariance, we get a good tool for pattern recognition because the coefficients do not depend on the position, the orientation of the size of the object. Certain coefficients and certain relation of the atom parameters would mean the presence of a concrete pattern in the analyzed image.

Energy conservation

When we have an infinite decomposition, the energy in the transformed domain and the energy in the space domain is the same, as we can deduce from Equation 3.11. As

$$\lim_{M \to \infty} R^M f = 0 \tag{C.7}$$

due to the exponential decreasing of the coefficients, when $M \to \infty$, 3.11 turns to:

$$\|f\|^2 = \sum_{m=0}^{\infty} |\langle R^m f, g_{\gamma_m} \rangle|^2 , \tag{C.8}$$

which mimics Parseval's equality for Fourier series.

Invertible

A complete Pursuit recovers a perfect version of the image:

$$f = \sum_{m=0}^{\infty} \langle R^m f, g_{\gamma_m} \rangle g_{\gamma_m}. \tag{C.9}$$

Thus the image f may be reconstructed from its MP coefficients, but if the decomposition is not complete we will not be able to get a perfect reconstruction, we will have a reconstruction error given by $R^M f$, where M is the number of coefficients used by the decomposition.

Bibliography

[1] A.J.Ahumada, C.H.Null,*Image quality: A multidimensional problem*, Digital Images and Human Vision, ed. A.B.Watson, pp.141-148, MIT Press, 1993.

[2] A.Aldroubi, K.Gröchenig, *Non-uniform sampling and reconsturction in shift-invarian spaces*.

[3] O. Al-Shaykh, E. Miloslavsky, T. Nomura, R. Neff, A. Zakhor, *Video Compression Using Matching Pursuits*, IEEE Transactions on Circuits and Systems for Video Technology, 17(2):123–143,Feb. 1999.

[4] C.C.Bisell,S.Mallat, *Matching Pursuit of images*, IEEE Proc. Int. Conf. Image Proc., volume I,53-56,Washington DC, Oct. 1995

[5] M.Brady and Z-Y.Xie, *Feature Selection for Texture Segmentation*, Advances in Image Understanding, K.Bowyer and N.Ahuja (Eds.), IEEE Computer Society Press, 1996.

[6] C.van der Branden Lambrecht, *Perceptual models and architectures for video coding applications*, PhD Thesis, EPFL, 1996.

[7] A.Bovik, M.Clark, W.Giesler, *Multichannel Texture Analysis Using Localised Spatial Filters*, IEEE Trans. Pattern Analysis and Machine Intelligence, 12, pp.55-73, 1990.

[8] J.F.Canny, *A computational approach to edge detection*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.8(6), 679-698, Nov. 1986.

[9] J.R.Casas, F.Marqués, P.Salembier, *Processament d'Imatge (PIM), transparències del curs*, Deptartament de Teoria del Senyal i Comunicacions, UPC.

[10] E.Chang, S.Mallat, C.Yap *Wavelet Foveation*, Technical Report, New York University, Sep.1999.

[11] B.Chaudhuri, N. Sarkar, *Texture Segmentation Using Fractal Dimension*, IEEE Trans. Pattern Analysis and Machine Intelligence, 17, 1, pp.72-77, 1995.

[12] R.Chellappa, S.Chatterjee, *Classification of Textures Using Gaussian Markov Random Fields*, IEEE Trans. Acoustic, Speech and Signal Processing, 33, 4, pp.959-963, 1985.

[13] Y.Chen, E.Dogherty, *Grey-Scale Morphological Granulometric Texture Classification*, Optical Engineering, 33, 8, pp.2713-2722,1994.

[14] R.R.Coifman, M.V.Wickerhauser *Entropy-based algorithms for best basis selection* IEEE Transations on Information Theory, vol. 38(2), 713-718, 1992.

[15] S.Comes, *Les traitements perceptifs d'images numerisées*, PhD Thesis, Université Catholique de Louvain, 1995.

[16] J.W.Cooley, J.W.Tukey, *An algorithm for the machine calculation of complex Fourier series*, Math. Computat., vol.19, pp.297-301, 1965.

[17] G.Cross, A.Jain, *Markov Random Field Texture Models*, IEEE Trans. Pattern Analysis and Machine Intelligence, 5, 1, pp.25-39, 1983.

[18] I.Daubechies, *Ten lectures on Wavelets*, CBMS-NSF Series in Appl. Math., SIAM, 1991.

[19] J.G.Daugman, *Two-dimensional Spectral Analysis of the Cortical Receptive Field profiles*, Vision Research, vol.20, pp.847-856, 1980.

[20] J.Daugman, *Uncertainty Relation for Resolution in Space, Spatial Frequency and Orientation Optimised by Two-Dimensional Visual Cortical Filters*, Journal of the Optical Society of America, 2, pp.1160-1169, 1985.

[21] G.Davis, S.Mallat, M.Avellaneda *Adaptative Greedy Approximations*, Constructive Approximation, Springer-Verlag, New York INC.1997.

[22] H.Derin, H.Elliot, *Modeling and Segmentation of Noisy and Textured Images Using Gibbs Random Fields*, IEEE Trans. Pattern Analysis and Machine Intelligence, 9, 1, pp.39-55, 1987.

[23] R.A.DeVore, B.Jawerth V.Popov, *Compression of wavelet decompositions* American Journal of Mathematics vol. 114, 737-785, 1992.

[24] R.A.DeVore, *Nonlinear Approximation*, Acta Numerica 51-150, 1998.

[25] O.Divorra Escoda, *Sparse routines for Full Search Matching Pursuit*, LTS-CVS, 2003.

[26] D.L.Donoho, *Unconditional bases are optimal bases for data compression and for statistical estimation*, Appl. Comput. Harmon. Anal., vol. 1, 100-115, 1993.

[27] D.L.Donoho, S.Chen, M.A.Saunders *Atom decomposition by basis pursuit*, Technical report, Dept. Stat., Stanford Univ., Stanford (CA), 1996.

[28] D.L.Donoho, X.Huo, *Uncertainty principles and ideal atomic decomposition*, IEEE Transactions on Information Theory, vol.47, 2845-2862, Nov. 2001.

[29] H.G.Feichtinger, A.Türk, T.Strohmer, *Hierarchical Parallel Matching Pursuit*, NUHAG-Numerical Harmonic Analisis Group, University of Vienna, Department of Mathematics.

[30] J.A.Ferweda, *Elements of Early Vision for Computer Graphics*, IEEE Computer Graphics and Applications, pp.22-33, Sep. 2001

[31] R.M.Figueras i Ventura, *Image Coding with Matching Pursuit*, MS Thesis, EPFL, Aug. 2001.

[32] R.M.Figueras i Ventura, P.Vandergheynst, *Matching Pursuit through Genetic Algorithms*, LTS Technical Report 01.02, July 2001.

[33] R.M.Figueras i Ventura, P.Vandergheynst, P.Frossard, *Evolutionary Multiresolution Matching Pursuit and its relations with the Human Visual System*, Proc. EUSIPCO, vol. 2, 395-398, Toulousse (France), Sept. 2002.

[34] R.M.Figueras i Ventura, O.Divorra Escoda, P.Vandergheynst, *Full Search Matching Pursuit*, LTS Technical Report, publication pending.

[35] P.J.Franaszczuk, G.K.Bergey, H.Eisenberg, *Matching Pursuit analysis of Mesial Temporal Lobe Complex Partial Seizures suggests increased nonlineal complexity at seizure conclusion*, Annual Meeting of the American Epilepsy Society, San Diego, California, Dec. 6-9, 1998.

[36] J.H.Friedman, W.Stuetzle, *Projection pursuit regression*, Journal of American Statistics Society, vol. 76,817-823, 1981.

[37] P.Frossard, P.Vandergheynst, R.M.Figueras i Ventura, M.Kunt, *A posteriori quantization of Progressive Matching Pursuit streams*, To appear in IEEE Transactions of Signal Processing.

[38] P.Frossard, P.Vandergheynst, R.M.Figueras i Ventura, *High flexibility scalable image coding*, Proc. VCIP, Lugano, July 2003.

[39] G.Fumarola, *Sbidibuda con ombra per scapelamentro a destra*, MS Thesis, EPFL, (Publication Pending), July 2003.

[40] D.Gabor, *Theory of communication*, J.IEE, 93:429-457, 1946.

[41] M.R.Garey, D.S.Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H.Freeman and Co., New York, 1979.

[42] M.Gharavi-Alkhansari, T.S.Huang, *A fast orthogonal matching pursuit algorithm*, Proc. ICASSP, p.1389-1392, Seattle (WA), May 1998.

[43] A.C.Gilbert, M.Muthukrishnan, M.J.Strauss, *Approximation of functions over redundant dictionaries using coherence*, 14th Annual ACM-SIAM Symphosium on Discrete Algorithms, Jan. 2003.

[44] R.Grinbonval, M.Nielsen, *Sparse representations in unions of bases*, Technical Report 1499, Institut of Recherche en Informatique et Systèmes Aléatoires, Nov.2002.

[45] R.Haralick, *Statistical and Structural Approaches to Texture*, Proc. IEEE, 67, 5, 786-804, 1979.

[46] D.Hasler, *Perspectives on Panoramic Photography*, Ph.D. Thesis, Audiovisual Communications Laboratory, EPFL.

[47] R.Heath, T.Strohmer, A.J.Paulraj, *On quasi-orthogonal signatures for CDMA systems*, Proc. 2002 Allerton Conference on Communication, Control and Computers, 2002.

[48] M.T.Heideman, D.H.Johnson, C.S.Burrus, *Gauss and the history of the FFT*, IEEE Acoustics, Speech and Signal Processing Magazine, vol. 1, pp. 14-21, October 1984.

[49] Jain, Anil K., *Fundamentals of digital image processing*, Englewood-Cliff, NJ: Prentice-Hall, 1989.

[50] P. K. Jimack, N. Touheed, *An Introduction to MPI for Computational Mechanics*, In Parallel and Distributed Processing for Computational Mechanics: Systems and Tools, ed. B.H.V. Topping (Saxe-Coburg Publications), pp.24-45, 1999.

[51] Joint Picture Expert Group, *JPEG Standard*, `http://www.jpeg.org`.

[52] Joint Picture Expert Group, *JPEG 2000 Standard*, `http://www.jpeg.org/JPEG2000.html`.

[53] L.K.Jones, *On a conjeture of Huber concerning the convergence of projection pursuit regression*, The Annals of Statistics, vol.15, No.2, p.880-882, 1987.

[54] B.Julesz, *Experiments in the Visual Perception of Texture*, Scientific American, 232, 4,pp.34-43,1975.

[55] L.Kaplan, C-C. Kuo, *Texture Roughness Analysis and Synthesis via Extended Self-Similar (ESS) Model*, IEEE Trans. Pattern Analysis and Machine Intelligence, 17, 11, pp.1043-1056, 1995.

[56] S.A.Klein, *Image quality and image compression: A psychophysicist's viewpoint*, Digital Images and Human Vision, ed. A.B.Watson, pp.73-88, MIT Press, 1993.

[57] S.A.Klein et al., *Seven models of masking*, Proc. SPIE, vol.3016, 13-24, San Jose, CA, 1997.

[58] A.Laine, J.Fan, *Texture Classification by Wavelet Packet Signatures*, IEEE Trans. Pattern Analysis and Machine Intelligence, 15, 11, pp.1186-1191, 1993.

[59] M.Levine, *Vision in Man and Machine*, McGraw-Hill, 1985.

[60] E.Maggio, *Matching Pursuit in Video Coding*, MS Thesis, EPFL, (Publication Pending), Sept. 2003.

[61] S.Mallat, *Multifrequency Channel Decomposition of Images and Wavelet Models*, IEEE Trans. Acoustic, Speech and Signal Processing, 37, 12,pp.2091-2110, 1989.

[62] S.Mallat, Z.Zhang *Matching Pursuits with time-frequency dictionaries*, IEEE Transations on Signal Processing, vol. 41(12), 3397-3415, 1993.

[63] S.Mallat, *A Wavelet Tour of signal processing*, Academic Press (1998), USA.

[64] B.Manjunath, R.Chellappa, *Unsupervised Texture Segmentation Using Markov Random Fields*, IEEE Trans. Pattern Analysis and Machine Intelligence, 13, 5, pp.478-482, 1991.

[65] A.Materka, M.Strzelecki, *Texture analysis methods - A review*, Technical University of Lodz, Institute of Electronics, COST B11 report, Brussels 1998.

[66] Morse, Bryan S., *Lectures of Signal Processing course*, Brigham Young University, `http://rivit.cs.byu.edu/550/W02/`.

[67] *MPI - The Message Passing Interface Standard*, `http://www-unix.mcs.anl.gov/mpi/indexold.html`, 2002.

[68] M.Nadenau, S.Winkler, D.Alleysson, M.Kunt, *Human Vision Models for Perceptually Optimized Image Processing - A review.*

[69] R.Neff, A.Zakhor, *Modulus Quantization for Matching Pursuit video coding*, IEEE Trans. Circuits and Systems for video technology, vol.10, 6, pp.895-912, 2000.

[70] H.Niemann, *Pattern Analysis*, Springer-Verlag, 1981.

[71] B.Olshausen, D. Field, *Sparse coding with an overcomplete basis set: A strategy employed by V1*, Vision Research, 37:33113325, 1997.

[72] B.Olshausen, D.Field, *Sparse coding with an overcomplete basis set: A strategy employed by V1*, Vision Research, 37:33113325, 1997

[73] Y.C.Patri, R.Rezaifar, P.S.Krishnaprasad, *Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition*, 27^{th} Asilomar Conf. on Signal, Systems and Comput., Nov. 1993.

[74] A.Pentland, *Fractal-Based Description of Natural Scenes*, IEEE Trans. Pattern Analysis and Machine Intelligence, 6, 6, pp.661-674, 1984.

[75] L.Peotta, L.Granai, P.Vandergheynst, *Very low bit rate image coding using redundant dictionaries*, Proceedings of SPIE 48th Annual Meeting, San Diego, August 2003.

[76] B.D.Rao, *Signal Processing with the sparseness constraint*, Proc. ICASSP, p.1861-1864, Seattle (WA), May 1998.

[77] H.Ridder, *Naturalness and image quality: Chroma and hue variation in color images of natural scenes*, Proc. SPIE, vol. 2411, pp. 51-61, San Jose, CA, 1995.

[78] A.Rosenfeld, J. Weszka, *Picture Recognition* in Digital Pattern Recognition, K. Fu (Ed.), Springer-Verlag, pp.135-166, 1980.

[79] A.Rosenfel and A.Kak, *Digital Picture Processing*, vol.1, Academic Press, 1982.

[80] J.A.Roufs, *Perceptual image quality: Concept and measurement*, Philips J. Res. 47(1):35-62, 1992.

[81] A.E.Savakis, *Evaluation of image appeal in consumer photography*, Proc. SPIE, vol. 3959, pp.111-120, San Jose, CA, 2000.

[82] J.Serra, *Image Analysis and Mathematical Morphology*, Academic Press, 1982.

[83] E.Simoncelli, W.T.Freeman, *The steerable pyramid: A flexible architecture for multi-scale derivative computation*, 2nd Int'l Conference on Image Processing Proceedings, vol.3, pp.444-447 Washington DC, Oct. 1995.

[84] E.Simoncelli, J.Portilla, *A parametric texture based model based on joint statistics of complex wavelet coefficients*, Int'l Journal of Computer Vision, pp.1-38, Dec. 2000.

[85] M.Strzelecki, A.Materka, *Markov Random Fields as Models of Textured Biomedical Images*, Proc. 20th National Conf. Circuit Theory and Electronic Networks 97, Kołobrzeg, Poland, 493-498, 1997.

[86] V.N.Temlyakov, *Weak Greedy Algorithms*, Adv. Comput. Math., vol.12, p.213-227, 2000.

[87] V.N.Temlyakov, *A criterion for convergence of Weak Greedy Algorithms*, Adv. Comput. Math., vol.17, p.269-280, 2002.

[88] J.A.Tropp, *Greed is good: Algorithmic results for sparse approximation*, Texas Institute of Computational Engineering and Sciences, Technical Report, Feb. 2003.

[89] P.Vandergheynst, P.Frossard, *Efficient representation by anisotropic refinement in Matching Pursuit*, Proc. ICASSP, vol.3, Salt Lake City (UT), May 2001.

[90] L.F.Villemoes, *Nonlinear approximation with Walsh atoms*, 1996.

[91] A.B.Watson. *Handbook of Perception and Human performance*, Vol.1, Sensory Processes and Perception, Chapter 6, Temporal sensivity. John Wiley ed., 1986.

[92] A.B.Watson et al., *Image quality and entropy coding*, Proc. SPIE, vol.3016, 2-12, San Jose, CA, 1997.

[93] J.Weszka, C.Deya, A.Rosenfeld, *A Comparative Study of Texture Measures for Terrain Classification*, IEEE Trans. System, Man and Cybernetics, 6, 269-285, 1976.

[94] I.H.Witen, R.M.Neal, J.G.Cleary, *Arithmetic encoding for Data Compression*, Communications of teh ACM, vol.30, pp.520-540, June 1987.

[95] S.N.Yendrikhovskij, *Perceptually optimal color reproduction*, Proc. SPIE, vol. 3299, pp. 274-281, San Jose, CA, 1998.

[96] Z.Zhang, *Matching Pursuit*, Ph.D. dissertation, New York University, 1993.