LINKÖPING STUDIES IN SCIENCE AND TECHNOLOGY
DISSERTATION NO. 909

ADAPTIVE IMAGE COMPRESSION WITH WAVELET PACKETS AND EMPIRICAL MODE DECOMPOSITION

ANNA LINDERHED

INSTITUTE OF TECHNOLOGY
LINKÖPINGS UNIVERSITET

DEPARTMENT OF ELECTRICAL ENGINEERING
LINKÖPING UNIVERSITY, SE - 581 83 LINKÖPING, SWEDEN

LINKÖPING 2004

Adaptive Image Compression with Wavelet Packets and Empirical Mode Decomposition

anna.linderhed@foi.se

Deptartment of Electrical Engineering
Linköping University, SE-581 83 Linköping, Sweden

ISBN 91-85295-81-7 ISSN 0345-7524

Printed in Sweden by LTAB, Linköping 2004, no. 844.

To My Husband
Håkan
and Our Children
Ulrika, Karl, and Matilda

Abstract

This thesis addresses the problem of using wavelet packets and empirical mode decomposition (EMD) for image compression. The wavelet packet basis selection algorithm is studied through an extensive experimental survey of the generated decomposition trees. We formulate the "triplet problem" for image compression as follows: How is the decomposition tree related to the image content, filter set and cost function? Our aim is to find an optimal basis for compression of images. Results are presented using test images from the Brodatz texture set. We also present a method to analytically calculate the cost of splitting a node, for a given signal model and filter, without actually performing the split.

A totally different approach to signal decomposition is the EMD. This is an adaptive decomposition scheme with which any complicated signal is decomposed into its intrinsic mode functions (IMF). The concept of EMD is extended to two dimensions to make it useful for image processing. The EMD and the sifting process to generate the IMFs are described. Different known and newly found difficulties with implementation of the method in two dimensions are highlighted and solutions are proposed. The method of variable sampling of the EMD, using overlapping blocks, is presented and the concept of empiquency is introduced to describe spatial frequency since the traditional Fourier-based frequency concept is not applicable.

Several ways to use EMD for image compression are examined and presented. The two-dimensional extension of the EMD is original as well as its application for image compression.

Preface

I started my stay in the Image Coding Group with my master thesis work on three-dimensional sound algorithms in an image coding project. This introduced me to the compression side of the image processing problem. Before that I only knew about traditional image processing such as image analysis and restoration.

I knew that if I could find a very good representation of the image, the compression would be easy. This is the very core of image compression but I was not satisfied with the way the algorithms at the time treated all images the same way. I wanted an adaptive representation.

In 1993 I attended a lecture by Professor Coifman and fell in love with the idea of wavelet packets and best basis. This was brand new, not only to me. The following years I spent with my babies, learning about image compression and wavelet theory, teaching signal theory at the university and struggling with my main question that no one could give me the answer to: Which one of all these possible best bases is the best for compression purposes? And why? Are wavelet packets the best way to represent the image for compression? During this time several good wavelet image compression algorithms were developed by others but the wavelet packets seemed to be of no good use.

By 2000 I nearly gave up my work towards a PhD thesis and left the Image Coding Group for a position at the Swedish Defence Research Agency.

Then it happened again! I fell in love with an idea.

In April 2001 Dr. Norden E. Huang presented his idea of Empirical Mode Decomposition at a wavelet conference. This was what I had been trying to do with wavelet packets. A totally signal adaptive decomposition. I had to expand the method to two dimensions for the use on images and then find out how to use it for compression, if possible.

So what motivated the work in this thesis?

Love.....and Curiosity.....

Anna Linderhed
September 2004

Acknowledgements

The wavelet part of this thesis was funded by the Image Coding Group at Linköping University while the EMD part of this thesis was conducted during my spare time, funded by my husband, and also partly funded by the Swedish Defence Research Agency, FOI.

Many thanks to my supervisor Dr. Robert Forchheimer for letting me do this my way and for many inspiring discussions. Thanks also to all my friends and previous colleagues at the Image Coding and Information Theory groups at the Dept. of Electrical Engineering at Linköping University, for good company and support. To all my friends and colleagues at the Swedish Defence Research Agency, thank you for believing in me and pushing me to finish this thesis. Special thanks to Lena Klasén and Stefan Sjökvist for the company at late hours and weekends doing our thesis writing together. Thank you also Dr. Robert Forchheimer, Dr. Lena Klasén, and Håkan Linderhed for careful proof reading of the manuscript. And also thanks to the anonymous reviewers of my papers for their valuable comments.

Most of all I thank my family, my dear husband Håkan and my children Ulrika, Karl and Matilda, for still loving me. Mummy's book is finally finished!!!!

I aningen
finns
sanningen

Björn Rosendal

Contents

Chapter 1

Introduction

This chapter gives a short introduction to the underlying theories that form the basis for the forthcoming chapters. It also includes the problem formulation and the outline of the work discussed in the thesis. The scientific contribution is summarized in the last section where possible applications are also discussed.

The basis for this research is the problem of electronic communication of images which includes storing of images and transmission over the Internet, telephone network or television channel. The framework is the communication model by Shannon [77] as viewed in Figure 1.1. This model shows how a message is communicated over a disturbed channel. The message is coded in the transmitter, the source coding and the channel coding can be treated separately following [77]. The source coding block can be further decomposed into its subblocks according to the scheme in Figure 1.10. In the first part of this thesis the work is concentrated on the problem of efficient representation of the image in order to make it easy for the source compression algorithms, while in the second part of this thesis the search for an efficient representation is complemented with different compression methods. We also restrict ourselves to work only with gray scale images.

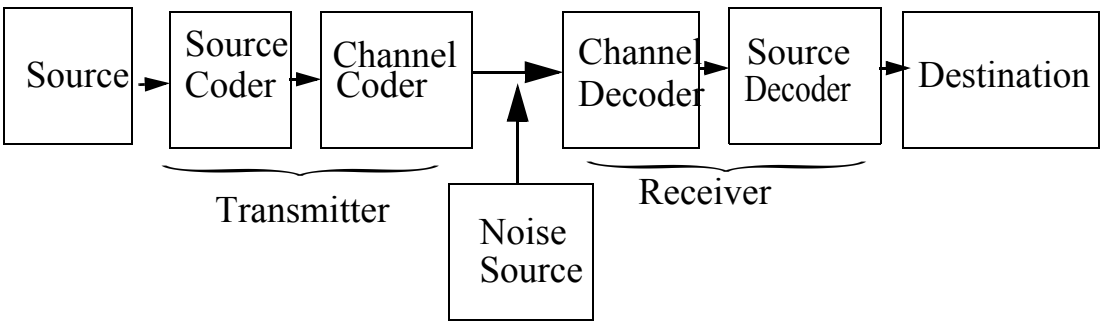


Figure 1.1. The Shannon communication model.

1.1 Properties of images

To be able to find an efficient representation of an image we have to understand the properties of images. An image can often be modelled as areas of texture, single or combined, and separated by edges. Some theories treat the set of image signals as a random process where the image is one realization of the random process, while others look upon the image as a deterministic signal. In this section we look at different ways to describe an image and its properties. Throughout this thesis we use the word *image* to denote the digital picture. The image is regarded as a square integer array in which each element is denoted a *pixel*. We ignore the digitizing process and assume all images we use are represented with 8 bits per pixel before our treatment starts. Pixel values thus range from 0 (black) to 255 (white).

1.1.1 Spatial frequency

Spatial frequency is a property that is often used for estimation of size and scale in images [30]. The images in Figure 1.2 show patterns of different spatial frequency along the horizontal direction.

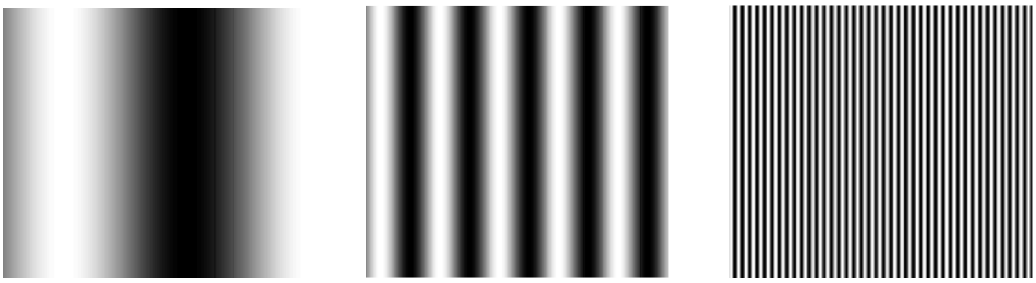


Figure 1.2. Patterns with different spatial frequency, a) low, b) medium, and c) high.

In a two-dimensional signal like an image the spatial frequency can have different directions. A spatial frequency component is traditionally described by

$$e^{j2\pi(ux+vy)} \tag{1.1}$$

which is a spatial pattern in which u and v are the spatial frequencies along the x- and y-axes [74]. The pattern has a spatial period of $(u^2+v^2)^{-1/2}$ along a direction that has an angle $\operatorname{atan}(u/v)$ with the x-axis.

Example 1.1. We analyse the spatial frequency in the horizontal direction of the image in Figure 1.3 by visual inspection of one row of the image.

Figure 1.3. Lenna 512x512 with row 450 marked in black.

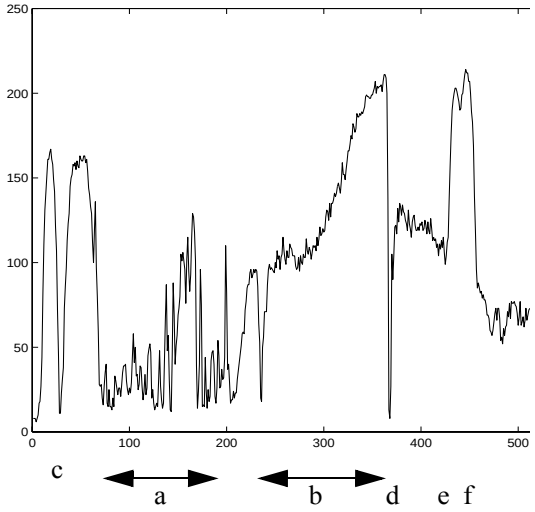


Figure 1.4. Row 450 of Lenna 512X512.

The fast varying values in Figure 1.4, part a, indicate high spatial frequency in the feather and the hair while the trend of slowly rising values in part b indicates the low spatial frequency in the shoulder. Edges are indicated by a dramatic change of value as in c, d, e and f in Figure 1.4.

1.1.2 Texture

Textures are homogeneous patterns or spatial arrangements of pixels that regional intensity or colour alone does not sufficiently describe. They are composed of a large number of more or less ordered similar patterns, giving rise to a perception of homogeneity [30]. The three principal approaches to describe the texture of a region used in image processing are statistical properties, structural properties, and spectral properties [29]. A texture may consist of the structured and/or random placement of elements, but also may be without fundamental sub-units. The properties that are important in the perception of visual

texture are complex visual patterns composed of subpatterns that have characteristic brightness, slopes, sizes, etc. The local subpattern properties give rise to the perceived lightness, directionality, coarseness, etc., of the texture as a whole [74]. In this thesis a *pure texture image* is characterized in that the whole image contains the same structure or a repeated pattern that the human eye/brain system interprets as a homogenous area of the same pattern all over the image.

1.1.3 Resonance

By spatial *resonance* we will mean the dominating repeated pattern, a periodic texture, in the image. An example of a resonance image is the brickwall texture. The dominant repeating structure is easily seen from the plot in Figure 1.6 of one row of the image in Figure 1.5.

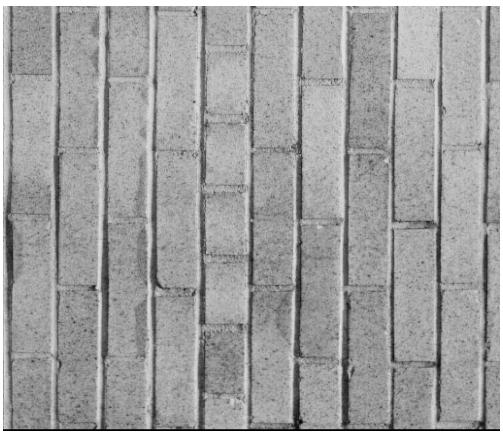


Figure 1.5. The brickwall texture.

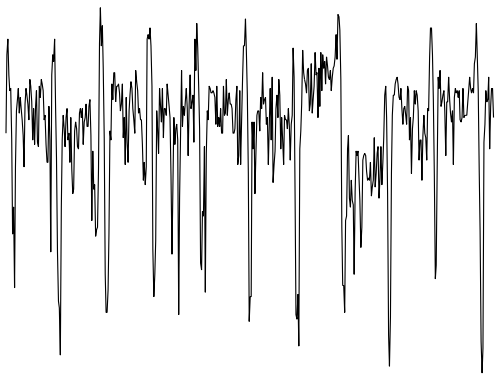


Figure 1.6. One row of the brickwall texture.

In Figure 1.4 we see one resonance in part a, the feather, and we see another resonance in part b, the shoulder.

1.1.4 Stationarity

Let X be the collection of all possible images. Following Rosenfeld & Kak [74], such a two-dimensional entity should be properly denoted *random field* corresponding to the one-dimensional random process. However, most of the time we will treat the images as one-dimensional vectors after a line scan in one of the directions. Alternatively we apply our algorithms in one direction at a time in a separable manner [30]. Thereforee we will use the notation *image random process* in this thesis. Any ensemble mean taken over the whole ensemble for one part of the image can be expected to be the same as the ensemble mean taken over the whole ensemble for another part of the image when the set includes all types of images. Also the autocorrelation function is not dependent on absolute location in the image thus and the process is *wide sense stationary* [22]. As a counter example of a set of images that is non-stationary in this sense we consider the set of face images used in video phone applications. Here all the images have a background and a face in the centre of the picture, a nose in the middle of the face, and shoulders and neck attached to the face. But in general we consider the images to be a weak sense stationary process and we can also talk about the spectral representation of our process.

We will talk about *deterministic non-stationarity* when there are different resonances in different parts of the image. In this sense a stationary image would be a pure texture image, such as the images treated in Chapter 3.

1.1.5 Spectral properties

Another way of describing the spatial frequency content of the image is to use the Fourier transform [9]. This is essentially the same as decomposing the image into its sinusoidal components. The frequency content of the process is contained in the *spectral density* [22] of the signal, also called *power spectrum* [74,64]. This can be calculated by taking the Fourier transform of the *autocorrelation function* [22] which describes the dependencies between pixels. The power spectrum can also be estimated by the *periodogram* [64] of the input signal.

When transforming the entire image we get no information on the location of the frequencies, and there is rarely a single clear resonance. When we transform only a small homogenous texture region of the image we get another result. This principle is used in the JPEG coding [65] of an image where the transform is taken over small parts of the image with the expectation that each part can be described by only a few frequency components.

Example 1.2. Compare the periodogram of the Lenna image in Figure 1.7 with the periodogram of the feather in her hat in Figure 1.8. The periodogram is taken over all the horizontal lines in the image together. The overall image is dominated by the low frequency components in the image while the high frequency components stands out when analysing only a high frequency part of the image.

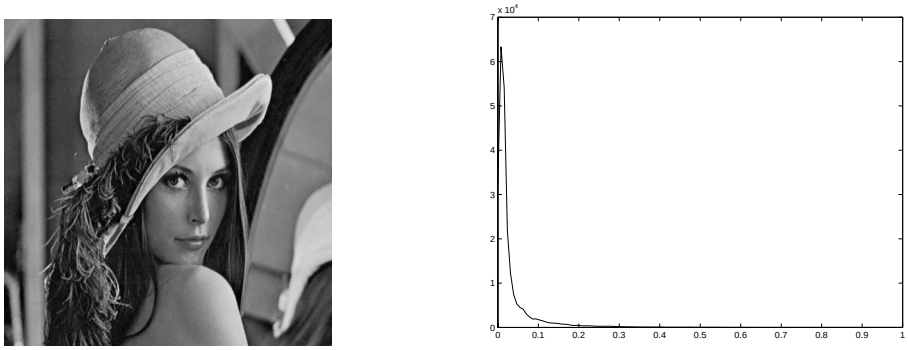


Figure 1.7. Periodogram of Lenna.

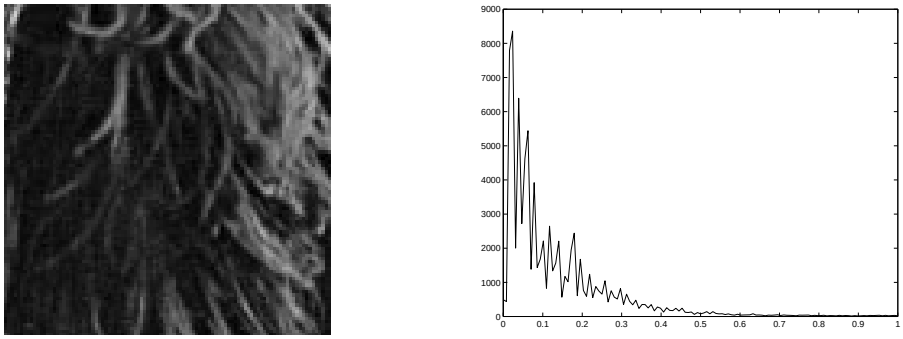


Figure 1.8. A small part of the feather in Lenna's hat and the periodogram of the same.

Example 1.3. The frequency content of the same part of the image can be different in different directions. Taking the periodogram of the brickwall texture in both the horizontal direction and the vertical direction shows the dominating low spatial frequency structure in one direction and the high spatial frequency resonance in the other direction, see Figure 1.9.

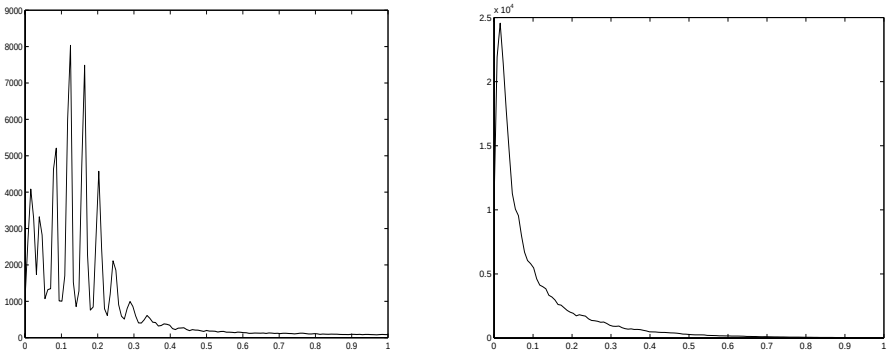


Figure 1.9. Horizontal and vertical periodogram of the brickwall texture.

1.2 The coding chain

Source coding involves not only the transform itself but also quantization, followed by

entropy coding. This is shown in Figure 1.10. If the signals to be encoded are realizations of a Gaussian process the design is well known. The basis set of optimal linear transforms for this signal is the Karhunen-Loeve (K-L) basis [40].

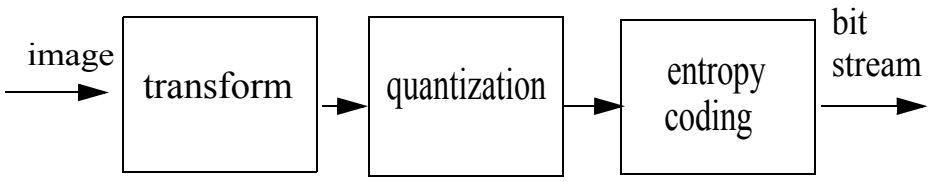


Figure 1.10. The coding chain.

1.2.1 Transform

The fast Fourier transform (FFT) algorithm [9] makes it possible to manipulate the image in the frequency plane by transforming the image. The decorrelation property of a transform is very important for compression efficiency. Compression is achieved when quantization is applied to the transform components.

The transform can be seen as a way to change the statistics of the source, a change of basis is a change of point of view. The transform $\bar{y}$ of a vector $\bar{x}$ is represented by a vector multiplication by a transform matrix T.

$$\bar{y} = T\bar{x} \tag{1.2}$$

After manipulations of $\bar{y}$, such as quantization, the reconstructed vector $\hat{\bar{x}}$ is achieved by

$$\hat{\bar{x}} = T^{-1}\hat{\bar{y}} \tag{1.3}$$

if the inverse T^{-1} exists.

A good decorrelating transform will remove *linear* dependencies from the data, thus producing a set of components such that, when individually quantized and entropy coded, the resulting symbol stream is reduced substantially, compared to applying the quantization directly on the image data.

The short time Fourier transform is the most widely used method for studying time varying signals. It is well understood and for many signals and situations it gives a good time-frequency structure, however, for certain situations it is not the best method.

The discrete cosine transform (DCT) established itself before the wavelet revolution and is one of the building blocks of the JPEG still image coding standard [65]. This transform is very close to the Karhunen-Loeve Transform (KLT) which produces uncorrelated transform coefficients of a Gaussian source. In the JPEG standard the two dimensional DCT uses a set of 64 8x8 tap basis functions shown in Figure 1.11. The image is divided into 8x8 pixel sized blocks and the transform components for each block found by correlation with the basis functions. This representation of images relates to the short time Fourier transform in that it analyses the image in fixed sized windows. Because the image is segmented into 8x8 pixel blocks, a basis function of a given spatial frequency can only

represent partial phase of a spatial oscillation within the block. The segmentation also gives rise to visible discontinuities between adjacent blocks, this is called the “blocking artifacts”. For some applications in this thesis we use sizes other than the usual 8x8 block.

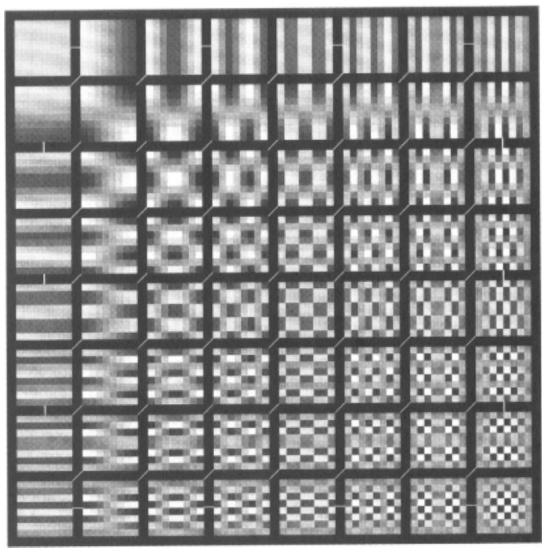


Figure 1.11. The set of 64 DCT basis functions used in the JPEG standard, from [65].

1.2.2 Quantization and bit allocation

The image transform used in a coder is usually invertible. That is, the image can be reconstructed from its transform coefficients. The quantization and thresholding stage however introduces errors, but also reduces the bitrate needed to represent the image. In this thesis uniform quantization is used except for some sections where SPIHT (see section 2.2.3) is used.

We use the mean square error (MSE)

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{x}_i)^2 \tag{1.4}$$

as distortion measure. The signal-to-noise ratio for any signal is

$$SNR_{dB} = 10\log\left(\frac{\sigma^2 + m^2}{MSE}\right) \tag{1.5}$$

For images represented with 8 bits per pixel (bpp) we use the peak-signal-to-noise ratio measure (PSNR).

$$PSNR = 10\log\left(\frac{255^2}{\frac{1}{N}\sum_{i=1}^{N}(x_i-\hat{x}_i)^2}\right) \tag{1.6}$$

1.2.3 Entropy coding

In most of the wavelet work in this thesis the bitrate is estimated by the entropy of the quantized components. However, in other parts of the thesis we will perform "real" coding using one of the following methods.

Huffman coding

The Huffman coder [18] assigns codewords to each component in relation to the probability of the value it holds. More probable values gets shorter codewords while rare values get long codewords. This way the components can be represented more efficiently.

Runlength coding

After quantization of the transform components many of them have the value zero. The length of zero runs can be coded separately. The simplest way to code the runlengths is to send a sequence of zeroes followed by a 1 (unary coding). 1 is coded as 01, 2 is coded as 001, 9 is coded as 0000000001 etc. Sorting the runlengths and coding the most frequent ones with the shortest codewords can give some improvement.

Elias code

Using Elias code [25] gives us an efficient way to assign codewords to the runlength values x. This code represents the integer x with its binary representation and a prefix telling the length of the binary representation of x. There are different ways to generate the prefix.

The simplest one is to use unary coding of the length of the binary representation giving codewords of length $1 + 2\lfloor \log(x) \rfloor$.

Using Elias code to code the length of the binary representation as well gives a code that maps the integer x to a codeword consisting of $1 + \lfloor \log(x) \rfloor$ bits followed by the binary value of x with the leading 1 deleted. The resulting codeword has length $\lfloor \log(x) \rfloor + 2\lfloor \log(1 + \lfloor \log(x) \rfloor) \rfloor + 1$. This gives shorter codewords for large values of x.

Coding of amplitude

For the coding of amplitude levels we will commonly use fixlength code of varying sizes.

1.3 Wavelet packet image coding

Image properties, filter and cost function make up a closely related triplet in selecting a wavelet packet basis. Changing any of these three affects the compression performance. Analysing the *triplet problem* is to understand *why and how the wavelet packet basis is affected by the choice of attributes image, decomposition filter and cost function*. In this thesis we analyse the triplet problem in a compression application. We use the expression *best basis* for the basis found by the best basis search algorithm of Coifman & Wickerhauser [17]. This algorithm uses a bottom up search strategy and is stated to find a global optimal basis. The expression *wavelet packet basis* is used for any non-redundant basis extracted from the full wavelet packet tree. In this thesis we are concerned mainly with the compression ratio achieved by the wavelet packet transform. We use texture images in this work because we are specifically interested in efficient representation of the resonance in the image.

Previous work used the fixed wavelet basis while discussing suitable filters, or the wavelet packet basis with discussion of cost functions but mostly without considering filters and image properties.

A triplet database consisting of coding results from 2295 different triplets has been created using the best basis algorithm. This data is analysed and discussed with respect to different images, cost functions, and filters.

An important question is whether, and when, a wavelet packet basis is better than the fixed wavelet basis. We will attempt to answer this question by combining the best filter and cost function with a flexible wavelet packet coder described in Chapter 2.

We present a method to analytically calculate the expected wavelet packet basis, with a top-down approach, for certain triplets using signal models to represent the image.

1.4 Empirical mode decomposition

A totally different approach to signal decomposition was first presented by Huang et al. [36]. Empirical mode decomposition (EMD) is an adaptive decomposition with which any complicated signal can be decomposed into its intrinsic mode functions (IMF). The Fourier spectral analysis as well as wavelet methods require the system to be linear and the data must be strictly periodic or stationary. EMD is an analysis method that in many aspects gives a better understanding of the physics behind the signals. Because of its ability to describe short time changes in frequencies that can not be resolved by Fourier spectral analysis it can be used for non-linear and non-stationary time series analysis.

In the second part of the thesis we will investigate how to apply EMD to images and image coding. Previous work often mentions the lack of a mathematical formalism to describe the EMD; the concept is truly empirical. In this work we keep the empirical approach as we extend the method to two-dimensional signals, such as images. We present an EMD method that can decompose the image into a number of IMFs and a residue with no, or only a few extrema points. This method makes it possible to use the EMD for image processing.

Next we examine the use of EMD for image compression purposes. The set of IMFs and the residue image is a very redundant way to represent an image. Our first goal is to represent the image by its EMD with the same number of samples as the original image.

Our second goal is to find an EMD-based image coder that can achieve reasonable compression. The concept of *empiquency*, short for empirical mode frequency, is introduced to describe the signal oscillations, since the traditional frequency concept is not applicable in this work.

1.5 Overview of the thesis and summary of publications

This thesis consists of two main parts. The first part, Chapters 2 to 4, deals with wavelet packet coding of images, an area where much research has already been done. However, in this part we focus specifically on how the choice of filters and cost functions influences the basis selection in relation to the images. The second part, Chapters 5 to 11, deals with the idea of using empirical mode decomposition for signal compression. This is new, as is applying EMD to images. Parts of the thesis work are included in the following reports:

- *J. Karlholm, M. Ulvklo, S. Nyberg, A. Lauberts, A. Linderhed, "A survey of methods for detection of extended ground targets in EO/IR imagery". FOI 2003, 144 pp. Scientific report FOI-R--0892--SE, Linköping*

in which the author of this thesis contributed with a discussion on image compression of sensor data.

- *J. Karlholm, M. Ulvklo, J. Nygårds, M. Karlsson, S. Nyberg, M. Bengtsson, L. Klasén, A. Linderhed, M. Elmqvist, "Target detection and tracking processing chain: a survey of methods with special reference to EO/IR sequences". FOA 2000, User report FOA-R--00-01767-408,616--SE, Linköping*

in which the author of this thesis contributed with the wavelet packet representation of texture.

Chapter 2 deals with image coding, wavelet and wavelet packet methods, and presents a survey of such image coding methods. The focus is on image compression methods that have had major impact on the development of wavelet and wavelet packet image compression.

In Chapter 3 the wavelet packet triplet problem is analysed to understand why and how the wavelet packet basis is affected by the image, decomposition filter and cost function.

This question arose early when different best bases occurred for the same image when using different filters or cost functions. The problem is analysed by the author in:

- *A. Linderhed, "Cost function and filter influence on wavelet packet decompositions of texture images". Proceedings of SSAB symposium on image analysis, SSAB'00, Halmstad, 7-8 March 2000, pp.5-8.*

The exhaustive search of the combination of 17 images, 9 cost functions and 15 different wavelet filters was presented in

- *A. Linderhed, "Wavelet packet decompositions of texture images- Cost function and filter influence" Proceedings of the Internal Workshop for Information Theory and Image Coding 2001, Report LiTH-ISY-R-2345, Linköping University, 2001, pp. 49-73.*

In this thesis we draw some conclusions from this work that explain the different bases and guides to a choice of filter and cost function. With this in mind, one of the wavelet packet coders from Chapter 2 is tested on different images. The test verifies the hypothesis that *for pure texture images a wavelet packet basis adapted to the actual image is better than a general wavelet basis*.

The chapter continues with an analytic calculation of the expected wavelet packet basis based on two simple signal models, filters and cost functions. Part of this work was presented in:

- *A. Linderhed, "Wavelet packet decompositions of texture images: analysis of cost functions, filter influences, and image models". SPIE Proceedings Vol. 4738, Wavelet and Independent Component Analysis Applications IX, Orlando, FL 2002, April 2002, pp.9-20.*

In this thesis the work is extended to show that the best basis can be found, for a selection of triplets, with a top down approach using calculations based on image statistics.

Chapter 4 presents a discussion of the first part of the thesis.

With Chapter 5 the second part of this thesis starts. This chapter focuses on empirical mode decomposition, and starts with a description of the original work and its purpose as we understand it. The EMD and the sifting process for the IMF are described. The EMD is extended to two-dimensional signals. This was first presented in:

- *A. Linderhed, "2D empirical mode decompositions in the spirit of image compression", SPIE Proceedings Vol. 4738, Wavelet and Independent Component Analysis Applications IX, Orlando, FL 2002, April 2002, pp.1-8.*

In this chapter different known and newly found difficulties with implementation of the EMD method in two dimensions are highlighted and solutions are proposed. We also introduce the concept of *empiquency,* published in [53], to describe the spatial frequency in the image since traditional the Fourier-based frequency is not applicable in this work.

In Chapter 6 we present different attempts to compress the EMD. The results from this work are only published in this thesis.

The EMD is a redundant representation of an image. To overcome this a method of variable sampling of the EMD, using overlapping blocks, is proposed in Chapter 7. This is published in:

- *A. Linderhed, "Variable Sampling of the Empirical Mode Decomposition of Two-Dimensional Signals", International Journal of Wavelets, Multiresolution and Information Processing, special issue on Sampling Problems Related to Wavelet Theory and Time-Frequency Analysis, forthcoming.*

In Chapters 8 to 10 we present different image coders using this approach. A variant of these is presented in:

- *A. Linderhed, "Image compression based on empirical mode decomposition". Proceedings of SSBA symposium on image analysis, SSBA'04, Halmstad, 11-12 March 2004, p. 110-113.*

The rest of the image coding methods in this chapter are only published in this thesis so far, although several papers are in preparation.

Chapter 11 presents a discussion of the EMD work and how the result relates to the work of others. This chapter also present some open problems in this area.

1.6 Scientific contributions

The contributions of this thesis are:

- Survey of image coding methods using wavelet and wavelet packet techniques,
- Analysis of the triplet problem,
- Discussion of filters suitable for wavelet packet decomposition from an experimental and theoretical view,
- Analytic calculation wavelet packet basis desicion rule,
- Extension of EMD to two dimensions,
- Concept of empiquency,
- Different methods to compress the EMD,
- Variable sampling of EMD,
- Image compression algorithms based on variable sampling in combination with entropy coding and a DCT coder.

1.7 Possible applications

The work presented in this thesis is strictly concerned with image coding applications. During the time frame of the PhD studies the author has also been involved in another type of work that may benefit from the results described here.

Humanitarian demining is an important task to support the population of a region affected by a conflict in returning to the normal use of the country, e.g. for agriculture or the construction of infrastructure facilities. In this context the EU funded ARC project aimed at supporting Mine Actions by providing a tool for the fast, accurate and cost-efficient mapping of a mine suspected area and for minefield area reduction. The project is presented in [87] and [42]. The major contribution by the author to this project is the idea and implementation of temporal signal processing, detecting modelled object signatures in a diurnal IR-image sequence. The temporal signal processing in [79,80,81,49] stems from the author of this thesis. Including EMD in the analysis may provide yet another approach in the search for better mine detection methods. The EMD method has already shown excellent classification performance in the analysis of laser vibrometry signals [61].

PART ONE

WAVELET PACKET IMAGE COMPRESSION

Chapter 2

Wavelet and Wavelet Packet Methods for Image Compression

The first part of this thesis starts with an introduction of the wavelet and wavelet packet concept and continues with a survey of different image coding methods using these methods that have evolved over the years. We will go through the history and literature and present these techniques that lead us on to the problem we have focused on in this thesis. The compression results given in this chapter are only the results reported in the reviewed papers.

2.1 Introduction

Hierarchical methods (pyramidal, multi-resolution) for representing images have several advantages over plain sampled images or the flat representation. The first to use hierarchical representation of images for the purpose of compression were Adelson & Burt [1]. In subband representation [94] of an image the subsampled approximation of the original image is produced in the same way as in the pyramid coding scheme, but instead of the difference image used in [1], the subband coding scheme produces the *detail image* by a filtering of the original image with a high pass filter. In this section we will introduce two variations of the subband coding scheme, namely the wavelet transform and the wavelet packet transform.

2.1.1 Wavelet transform

The wavelet transform of an image is also a multiresolution description. The possibility

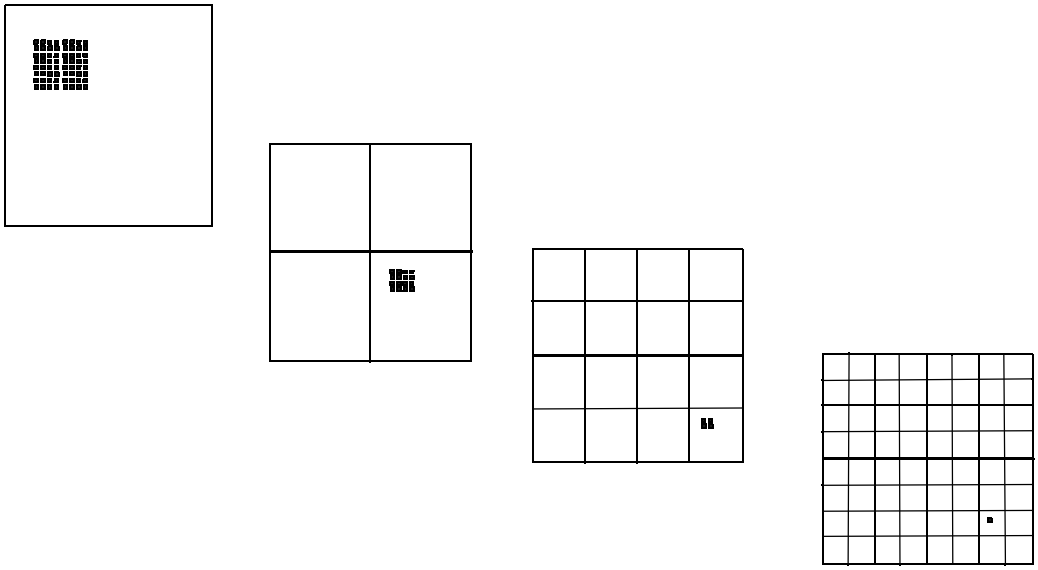
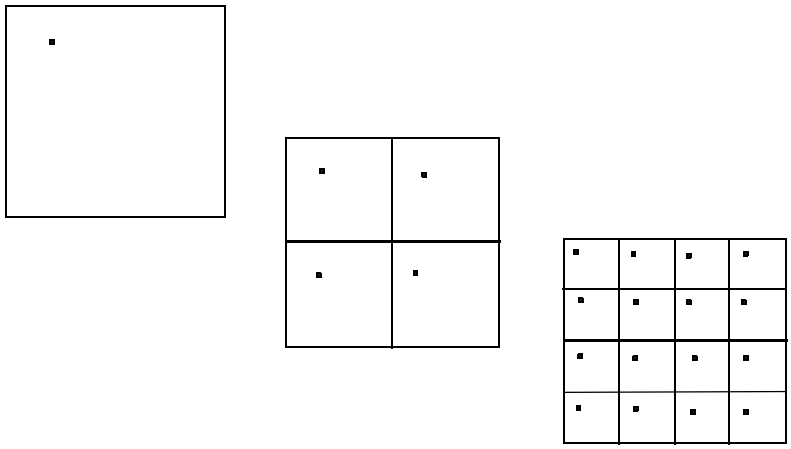


Figure 2.2. Components correspondence between subbands on different levels starting with one component in the lowest subband (right).

of using the wavelet theory in image compression was outlined in Mallat [54] where the multiresolution approach used in the subband coding schemes and the wavelet theory was merged. The wavelet theory introduced new nomenclature into the subband technique because it wed together different disciplines such as mathematics, seismology, physics and signal processing into the same field. The difference compared to subband coding is that the wavelet theory has equipped us with a tool for proper filter construction [20]. The wavelet transform uses scaled and translated versions of a prototype wavelet as basis functions to represent a signal. For details of the wavelet theory, the reader is referred to the text books [19,56,90,93].

The high-pass filtered data set is the wavelet transform *detail* components at that level of scale of the transform. The low-pass filtered data set is the *approximation* components at that level of scale. Thanks to the subsampling, the four sets of components have four times fewer elements than the original data set. The approximation components can now



·e 2.1. Component correspondence between subbands on different levels, starting with a pixel in the original image.

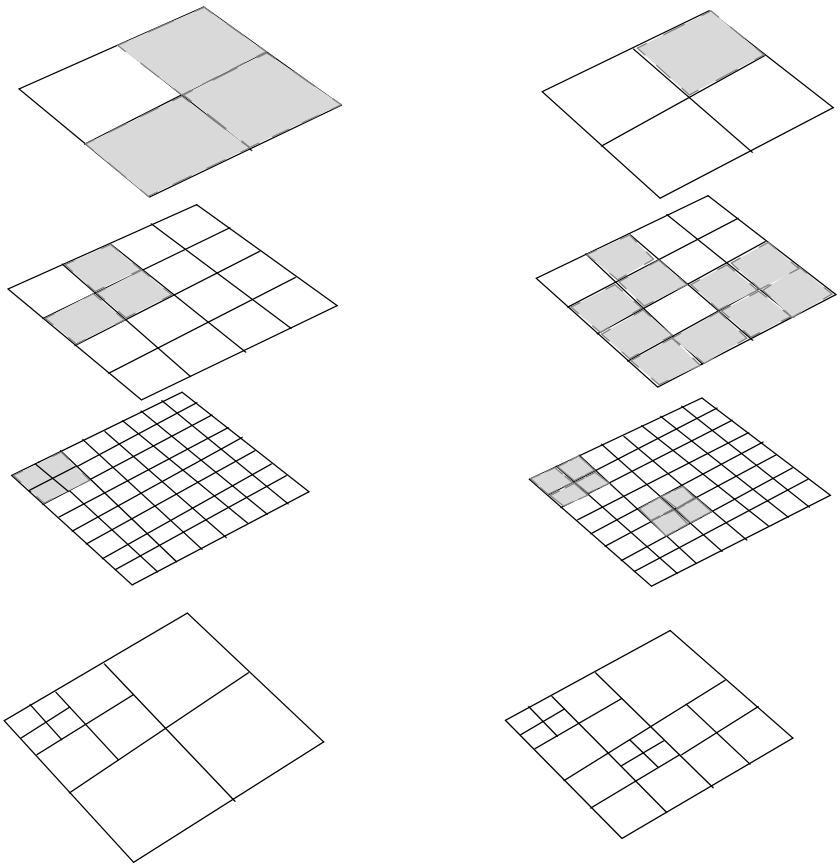


Figure 2.3. The subband stack. Subbands of different scales, the chosen subbands in gray form the resulting orthogonal basis shown at the bottom, to the left the wavelet basis and to the right a wavelet packet basis.

be used as the sampled data input for another pair of wavelet filters, identical to the first pair, generating another set of detail and approximation components at the next lower level of scale. The wavelet theory is an impressive mathematical tool to produce proofs that this really produces a representation of the image, proofs that did not exist in the time of subband theory. Today, wavelet coding has become the substitute name for essentially any kind of subband coding of images. This thesis includes a survey of wavelet and wavelet packet image coding methods.

Another way of looking at the representation is by looking at the subband correspondence of components. Due to the subsampling the size of the data matrix is the same for all levels but higher levels contain more subbands. One component in a higher subband represents the spatial frequency in an area of several pixels in the original image, having the same size and location as the component would cover if the subband is upsampled to the original image size. This is illustrated in Figure 2.2. A particular pixel in the original image is represented in every subband, indicating the amount of spatial frequency of the subband at the location of the pixel as shown in Figure 2.1.

2.1.2 Wavelet packets

Since texture typically is a composition of mid and high spatial frequency components

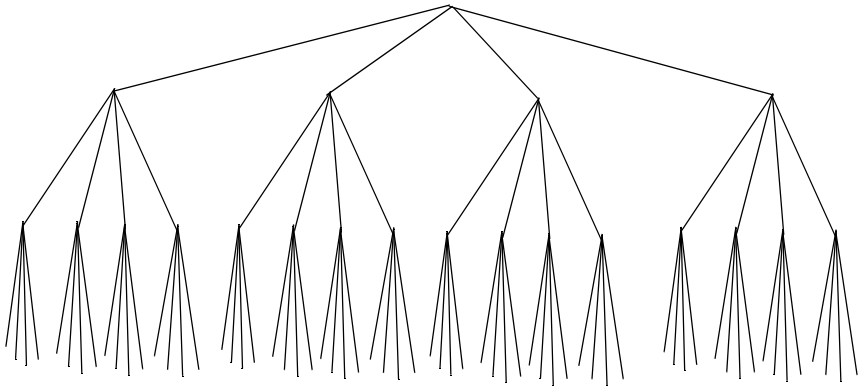


Figure 2.4. Fully decomposed 3-level wavelet packet tree.

the wavelet decomposition does not capture its typical structure. These structures are better found by the wavelet packet algorithms. The wavelet packet technique was introduced in [17] as a natural extension of the wavelet techniques and was immediately investigated in various areas of signal analysis, [70,45,66] among others.

The wavelet packet basis is constructed adaptively based on some cost function and choice of decomposition filter applied to an image. The decomposition of a signal can be viewed as a tree, where the left branch represents the low-pass horizontal/low-pass vertical filtering, the right branch represents the high-pass horizontal/high-pass vertical filtering and the middle branches represents the low-pass horizontal/high-pass vertical filtering and the high-pass horizontal/low-pass vertical filtering, respectively.

The wavelet packet tree is constructed in two steps. First, the image is filtered and subsampled into four new images representing the spatial frequency subbands. Each of the subbands are filtered and subsampled into four new images, a process that is repeated to a certain level. By keeping the components in every subband at every level the wavelet packet tree has the advantage of attaining the complete hierarchy of segmentation in frequency and is a redundant expansion of the image. This is illustrated by the subband stack in Figure 2.3. In the second step we can choose our best basis to represent the texture by cutting off branches in the tree controlled by the cost function applied on one node and on its children nodes. These subbands are to be seen as a collection of bases for this particular image and filter choice. Choosing a best basis is a way to represent the image in the most effective way.

The fully decomposed wavelet packet (WP) tree is shown in Figure 2.4. The wavelet packet transform generates an orthogonal basis if a proper wavelet is chosen.

For basis selection we apply a cost function on the unsplit subband *A, M(A)*, in Figure 2.5, and the same cost function on the subbands *B, C, D* and *E* resulting from the one level subband decomposition.

Split the subband if $M(A) > M(B) + M(C) + M(D) + M(E)$, otherwise keep the subband *A* as it is.

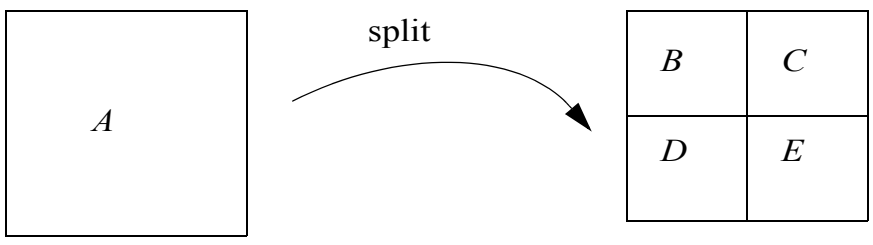


Figure 2.5. One step subband split.

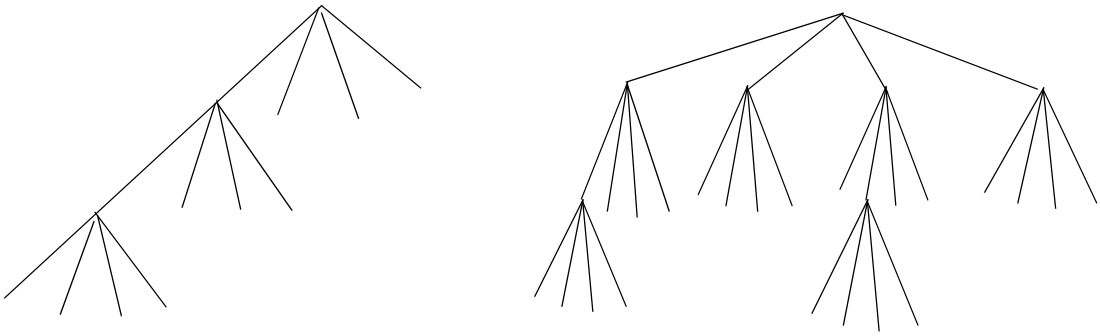


Figure 2.6. Tree representation of the bases in Figure 2.3.

In Figure 2.6 are shown two pruned trees, the left representing the wavelet basis and the right representing a possible best basis. The leaves of the tree represent the basis. Below are shown some texture images from the Brodatz collection [10] and their best bases, using a certain decomposition filter and cost function, specified in section 3.1. Wavelet packet decomposition of an image and basis selection results in different bases when using different filters or different cost functions. The best basis found for a specific image, using a particular filter and cost function, is best for this triplet of image, filter and cost function. Changing at least one of the three may result in another best basis. In Figure 2.7 the same filter and cost function are used on three different images.

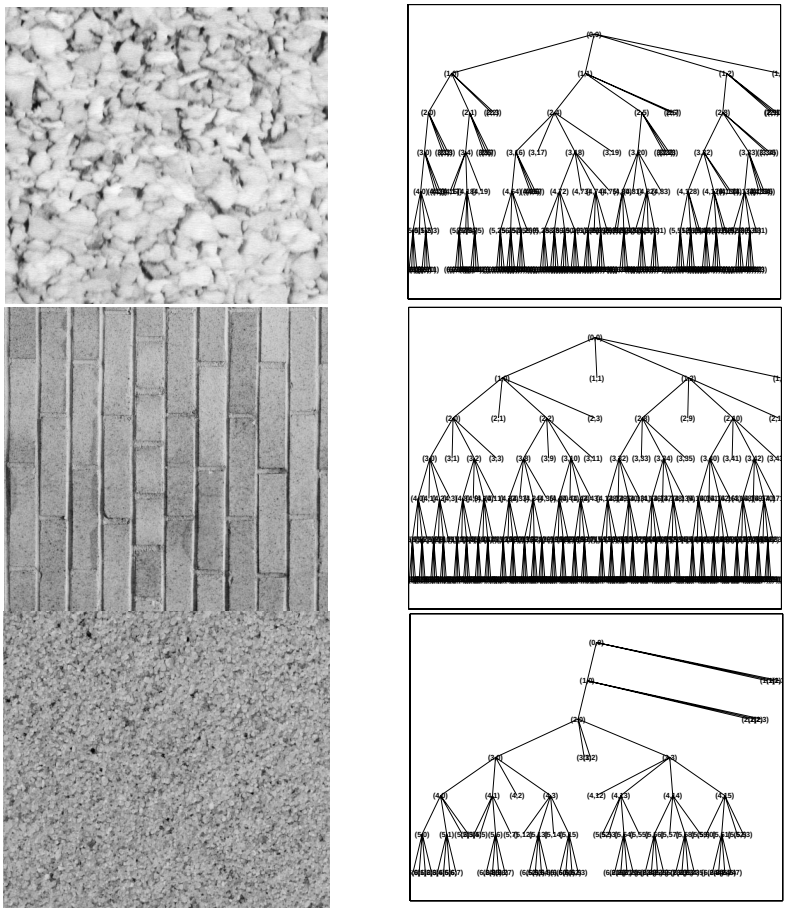


Figure 2.7. Texture images and their best wavelet packet trees.

2.2 Wavelet image coding algorithms

Wavelet transform coders are mostly used at rates below 1 bpp and are able to recover images at nearly perfect visual quality. Already from the beginning of the wavelet theory, the method was considered for image compression [3,55], due to the close connection to subband coding [94].

2.2.1 Wavelet coding with vector quantization

The coding algorithm of [3] uses the wavelet decomposition together with a multiresolutionvector quantization scheme and a noise shaping bit allocation procedure as explained below. The image is decomposed into its wavelet basis using a set of biorthogonal filters [19].

The vector quantization procedure was already accepted in the subband community as a powerful tool for image compression. The principle involves encoding a sequence of samples (a vector) rather than encoding each sample individually. Encoding is performed by approximating the sequence to be encoded by a vector belonging to a codebook. A multiresolution codebook is created and optimized using a classifying algorithm with a mean square error criterion; it contains vectors for each resolution level and direction. In Antonini et al. [3] the training set is a set of vectors belonging to different images including the image to be coded.

The bit allocation is based on the fact that the human eye is not equally sensitive to all spatial frequencies. The bit allocation, which is a function of the image, is transmitted as side information. For the coding of the 256x256 Lenna image the bit allocation scheme is shown in Figure 2.8. In Table 2.1 the coding performance of the Lenna image with different biorthogonal filters is shown.

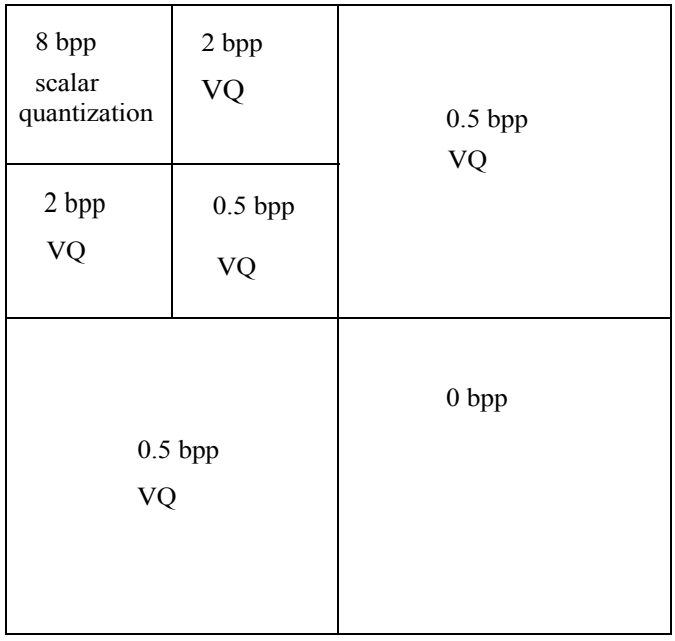


Figure 2.8. The bit allocation of the wavelet coder.

	PSNR		
Bitrate	**9-3 filter**	**5-7 filter**	**9-7 filter**
0.8 bpp	31.82dB	31.46 dB	32.10 dB

Table 2.1. Coding performance of the Lenna image with different biorthogonal filters.

2.2.2 Embedded zerotree wavelet algorithm (EZW)

For transform coders operating below 1 bpp with optimized scalar quantization and bit allocation the performance depends mostly on the encoding efficiency of the position of the zero components. Also an important part of the low bitrate image compression is the coding of the amplitude of those components that will be transmitted as nonzero values.

The zerotree was defined in order to improve the compression of the significance map in which only the components which are significant according to a specified threshold are represented. The idea of coding the significance map as a zerotree was introduced by Lewis & Knowles in [46] and beautifully developed by Shapiro in [78] to become an embedded image coder. The zerotree is based on the hypothesis that if a wavelet component at a coarse scale is insignificant with respect to a given threshold T, then all wavelet components of the same orientation in the same spatial location at finer scales, the "children", are likely to be insignificant with respect to T.

A wavelet component a is said to be insignificant with respect to a threshold T if $|a| < T$. Such a component is quantized to zero. The others are significant components. Coding the positions of the significant coefficients is the same as storing a binary significance map b defined by

$$b = \begin{cases} 0 & if \quad |a| \leq T \\ 1 & if \quad |a| > T \end{cases} \tag{2.1}$$

Furthermore, a component a is said to be an element of a zerotree for threshold T if itself and all of its descendants are insignificant with respect to T. An element of a zerotree for threshold T is a zerotree root if it is not the descendant of a previously found zerotree root for threshold T.

The EZW algorithm [78] is a simple but effective compression algorithm that has the property of generating the bits in the bitstream in order of importance, yielding a fully embedded code. With an embedded bit stream the reception of code bits can be stopped at any point and the image can still be reconstructed. The algorithm is based on four key concepts, 1) a discrete wavelet transform, 2) zerotree coding of the components, which provides a compact multiresolutionrepresentation of significance maps, 3) entropy coded successive-approximation quantization, and 4) arithmetic coding.

Two separate lists of coordinates of the wavelet components are maintained. The dominant list contains the coordinates of the components that have not yet been found to be significant. The subordinate list contains the magnitude of the significant components. The scanning of the components during a dominant pass is performed in such a way that

no child is scanned before its parent. During the scan, each component that is not marked insignificant by another zerotree is coded with one of four symbols; 1) zerotree root, 2) isolated zero, 3) positive significant, and 4) negative significant. Each time a component is coded significant its magnitude is appended to the subordinate list. The string of symbols is then encoded using an arithmetric coder. The performance of the EZW coder, using a 6 level QMF pyramid decomposition, for two different test images is shown in Table 2.2.

	PSNR	
Bitrate	**Lenna**	**Barbara**
1.0 bpp	39.55 dB	35.14 dB
0.5 bpp	36.28 dB	30.53 dB
0.25 bpp	33.17 dB	26.77 dB
0.125 bpp	30.23 dB	24.03 dB
0.0625 bpp	27.54 dB	23.10 dB
0.03125 bpp	25.38 dB	21.94 dB
0.015625 bpp	23.63 dB	20.75 dB

Table 2.2. The performance of the EZW coder, using a 6 level QMF pyramid decomposition, for two different test images.

2.2.3 Set partitioning in hierarchical trees (SPIHT)

SPIHT was introduced in [75] as a refinement of the EZW algorithm. In SPIHT the way subsequent components are partitioned and the way the significance information is conveyed are fundamentally different than from EZW. SPIHT uses the same hypothesis as EZW that if a parent component is below a certain threshold then it is likely that all its descendants are below the threshold too. If this prediction is successful then SPIHT represents the parent and all its descendants with a single symbol called a zerotree, as in section 2.2.2.

SPIHT works in two passes, the ordering pass and the refinement pass. One of the main features is that the ordering is not explicitly transmitted. Another important fact is that it is not necessary to sort all the components. A component $a(i,j)$ is said to be significant with respect to a threshold if $|a(i,j)| > 2^n$ for a given *n*. The sorting algorithm divides the set of components into partitioning subsets T_m and performs a magnitude test

$$\max_{i,j \in T_m} \{|a(i,j)|\} \geq 2^n \tag{2.2}$$

If the subset T_m is insignificant, all components in T_m are insignificant. Otherwise the subset is significant and is partitioned again into new subsets $T_{m,l}$, and the procedure is repeated. The partitioning uses the hierarchy of the wavelet decomposition.

In the refinement pass the quantization of components is refined. SPIHT maintains

three lists of coordinates of components in the decomposition. These are the List of Insignificant Pixels (LIP), the List of Significant Pixels (LSP) and the List of Insignificant Sets (LIS). The LIP contains coordinates of components that are insignificant at the current threshold. The LSP contains the coordinates of components that are significant at the same threshold. The LIS contains coordinates of the roots of the spatial parent-children trees.

The efficiency of SPIHT is heavily dependent on finding zerotrees in the LIS, yet producing an embedded bit-stream code. The performance of the SPIHT coder with arithmetric coding of the bit stream is shown in Table 2.3.

bitrate	**PSNR**
0.5 bpp	37.2 dB
0.25 bpp	34.1 dB
0.15 bpp	31.9 dB

Table 2.3. Performance of the SPIHT coder with arithmetric coding of the bit stream, applied to the Lenna image.

2.2.4 JPEG2000

JPEG2000 is a recently (2000) developed coding standard for still images. The use of wavelet compression is one of its many features. Here we will only review the use of the wavelet coding and leave the rest of the standard, which is well described in [13] for the interested reader. The standard is working on image tiles of size 128x128 or 256x256, as if they were individual images, and the wavelet decomposition is applied to each tile as shown in Figure 2.9.

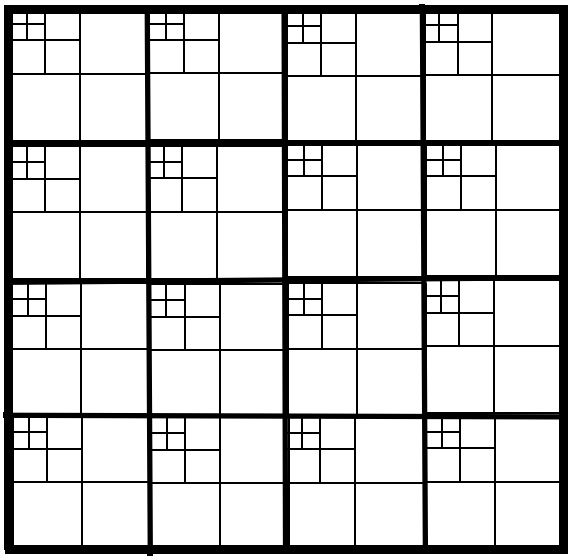


Figure 2.9. Wavelet decomposition of the tiles in JPEG2000.

The wavelet transform uses the 9-7 tap biorthogonal filter and Swelden's lifting technique [83]. The result of coding the Woman image using JPEG2000 with all its features is shown in Table 2.4.

bitrate	PSNR
1 bpp	41 dB
0.5 bpp	38 dB
0.25 bpp	35 dB

Table 2.4. Performance of the JPEG2000 coder applied to the Woman image.

2.2.5 Morphological representation of wavelet data (MRWD)

The goal of Servetto et al. [76] in their morphological representation of wavelet data is to find an improved statistical description of image subbands, capable of explicitly taking into account the space-frequency localization properties of the wavelet.

This alternative approach to the zerotree is based on a *region growin*g technique, where the significance map is encoded using local probability models depending on the region being encoded. If a given component is known to be classified as significant, then components in a small neighbourhood, with high probability, will be significant too (likewise for the insignificant ones). In a sense, the morphological approach proposed in [76] is a concept dual to that of zerotrees, in that it exploits *directly* the clustering of significant components. Also, arbitrarily shaped regions of nonzero components are defined at no extra cost.

The coder is a pure waveform coder. The morphological sets considered are not intended to capture any image domain objects or to exploit any property of the human visual system, but instead are tools used to assign probabilities to sets of wavelet components.

The probability describing intraband dependencies is determined, for each subband, by starting with a map of components all labelled insignificant, followed by computing a partition of components labelled significant or insignificant as follows.

Select a component at a random location within the current subband. If this selected component is not significant, or if it is significant but it has already been labelled significant, select another component at a random location until an unlabelled significant component is found. Then apply one step of morphological dilation at this location, and label all the dilated components significant. The morphological dilation is described in [76] as follows. Consider a set S to which the dilation operation will be applied, and let B represent some structuring element. Let $\oplus$ denote the morphological dilation operator. The dilated set $S \oplus B$ is defined to be the union of all points falling under the support of the structuring element B, when centred at each point in S. For each new significant components found by the dilation, recursively repeat this step, until no new significant components are found. Stop selecting components at random when all the significant ones have been labelled significant. Those without a label are labelled insignificant.

Define two histograms, corresponding to the frequency of occurrence of the components labelled significant and insignificant respectively by this partitioning method, and estimate the entropy of the composite model.

The encoder then scans subbands in raster scan order until a significant component is

detected. Then the encoder signals this event to the decoder with a special symbol, as side information. One step of dilation is performed: the encoder sends to the decoder and labels significant those coefficients in a neighbourhood of the current significant one. These new ones are examined: if new significant components are found, the process is applied recursively on each of these new values; otherwise, the process stops and raster scanning of insignificant components resumes, until the whole subband is exhausted. As a result, clusters of significant components are efficiently captured by morphological dilation: the only side information required to encode a new cluster is the symbol used to signal the transition from an insignificant to a significant region.

The resulting symbol stream is compressed using arithmetic coding based on the separate probability tables from above. The performance on the 512x512 Lenna image, using a five-level wavelet decomposition based on the spline 9-7 biorthogonal filters, with a uniform quantizer is shown in Table 2.5.

Bitrate	**PSNR**
1 bpp	40.33 dB
0.75 bpp	38.98 dB
0.5 bpp	37.17 dB
0.25 bpp	34.25 dB
0.125 bpp	31.09 dB
0.0325	25.82 dB

Table 2.5. The performance of MRWD on the 512x512 Lenna image.

2.3 Wavelet packet coding algorithms

The idea behind the wavelet packet coder is that the adaptive structure will lead to better coding results. The wavelet packet tree is any sub-set of a fully decomposed tree of a certain depth. To find the basis (sub-tree) a cost function is introduced. The tree is then searched for the basis that minimizes the selected cost function. If the filter and cost function are appropriately chosen the decomposition produced by the wavelet packet tree should, theoretically, always be better than that produced by the wavelet basis since this basis is included as a special case of the admissible wavelet packet structures.

2.3.1 Best basis subband coding

Although the wavelet packet and best basis concept were revealed to the public in 1992 [17] work had been done on images using this technique before that under the name of best basis subband coding [92]. It is based on the work of [16], at the time unpublished, extending it to two dimensions.

The best basis scheme avoids an exhaustive search over all bases. The algorithm starts by growing a full tree to the desired depth. Then, starting from the leaves, for every node the cost of keeping the node and pruning it is compared. The node is kept if the cost of keeping it is lower then that of pruning it. The process is continued at every level until the

root of the tree is reached.

In [17] QMF filters were used and the cost function denoted *energy entropy* (Equation 3.12). Compression is achieved by using a threshold ε on the components from the best basis and simply discharging the components below this threshold. Alternatively, for a fixed compression rate, the components are sorted and the appropriate number of components are used. Note that no quantization is applied to the components.

2.3.2 Rate-distortion optimized wavelet packet coding

The problem of optimal bit allocation in a wavelet packet framework was addressed in Ramchandran & Vetterli [69,70]. They used a rate-distortion criterion to find the best decomposition for a given signal. The idea is that each node in the fully decomposed tree is assigned the Lagrangian cost

$$L = D_i + \lambda R_i \tag{2.3}$$

Then the basis is chosen by forming the subtree which has the minimum sum of cost for its leaves. The decision to prune is based on the Lagrangian cost function for a fixed value of λ. The value of λ is found by sweeping through the sequence of values and for each value of λ the least cost quantization for each node is selected as the one that minimizes Equation 2.3. The total rate $\sum R_i$ is compared to the budget rate R_T, if $\sum R_i > R_T$ then λ is adjusted and the procedure starts over. The sequence of λ is achieved by using Newton's method. The basis selected should minimize the distortion given a bit budget, which is the essence of image coding.

In [70] the image is decomposed using the Db4 filter and trees were grown to a depth of 4 for a 512x512 image. First order entropy is used as the bitrate measure as scalar quantization is assumed, MSE is used as a distortion criterion. This codes the Barbara image to PSNR 36.4 dB at 0.92 bpp. The main achievement with this work was that it showed that the wavelet packet best basis approach performs better on high frequency textures than traditional coders.

2.3.3 Optimal entropy constrained lattice vector quantization (ECLVQ)

One of the advantages of wavelet packets for image compression is that it is adapted to the spatial frequency characteristic of the image. A wavelet packet basis can be chosen to maximize the energy compaction in each subband. In [43] wavelet packet representation is applied for entropy constrained image compression using lattice vector quantization. In resolving the mutual dependencies of the wavelet basis and source coding a two phase algorithm is used. First the optimal entropy constrained lattice vector quantization is developed for a given wavelet packet structure, then a greedy heuristic search algorithm is adopted to identify a locally optimal wavelet packet basis. The greedy algorithm does not guarantee an optimal basis; still, it performs as well as the other coders described here. Performance of the coder on the 512x512 Lenna image is presented in Table 2.6.

Bitrate	PSNR
0.5 bpp	36.97 dB
0.25 bpp	33.83 dB
0.15 bpp	31.82 dB

Table 2.6. Performance of the ECLVQ coder on the 512X512 Lenna image.

2.3.4 Constrained wavelet packet coding (CWP)

In this section a more thorough description of a SPIHT-WP-based coder is presented from Khalil et al. [41]. Especially the modifications necessary for using SPIHT with the adaptive tree structure are explained. The coder is used in the experiments in section 3.3 as a comparison with the more theoretical results.

With the wavelet-transform-based coder the octave band decomposition structure was fixed. The strength of the SPIHT algorithm for coding wavelet components is obvious. It would be desirable to combine the basis selection from the wavelet packet transform with SPIHT.

The parent-children relationship for SPIHT assumes that the decomposition structure is an octave-band decomposition. This means that in order to get SPIHT working with the more general wavelet packet structure (WP-structure) some modifications must be made.

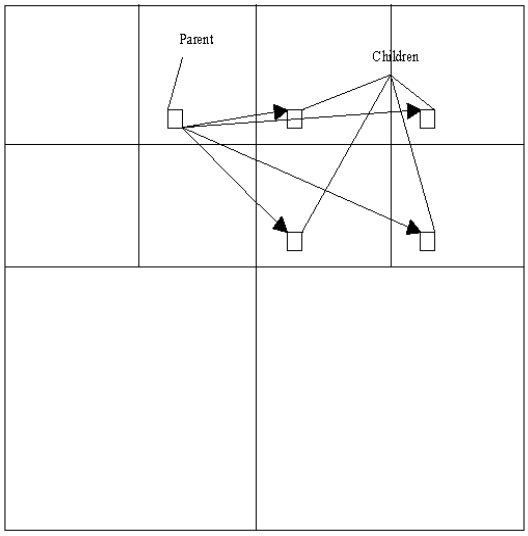


Figure 2.10. Parent-children relationship for a WP-structure.

In Figure 2.10 the region Parent is spatially co-located with the regions marked Children. In the wavelet transform case each child occupies a size corresponding to ¼ of that occupied by the parent. In the WP-structure in Figure 2.10 each child-region is at the same scale as the parent and thus occupies a region of the same size as the parent. The reason for choosing Parent as parent is that Parent is in a subband that has more low-pass character than the ones denoted Children, and low-pass subbands better predict the energy of high-pass subbands than the other way around.

Spatial reordering

To be able to use SPIHT efficiently on the WP-structure we have to take into account these new definitions of parent-children relationships, where children may reside in different subbands of the same scale. The modifications should be made so that inter-band similarities can be exploited. In Figure 2.11 this is exemplified for the same structure as in Figure 2.10.

The reordering permits the use of the SPIHT-algorithm without any changes to the SPIHT-algorithm. The simple 1:4 (two-dimensional case) parent-children mapping of a wavelet structure does not always hold in a general WP-structure. Instead the WP-decomposed data is reordered with some constraints explained below. In Figure 2.11 the simple 1:4 (two-dimensional case) parent-children mapping holds; in a general WP-structure this may not be the case. In Figure 2.12

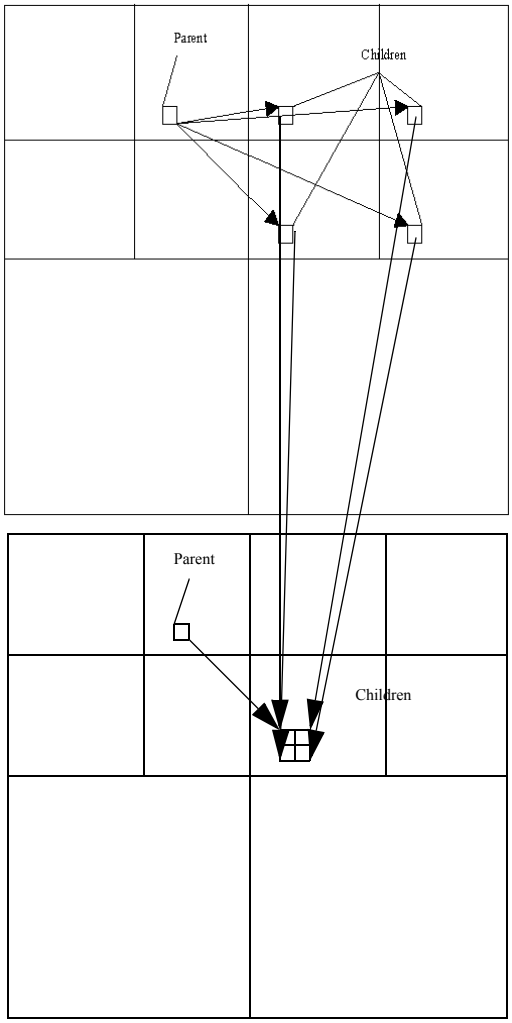


Figure 2.11. The upper figure is the WP-structure with the desired parent-children relationships. In the lower figure the data has been reordered so that the simple parent-children relationship hold.

Parent have 7 children of which four represent a region 4 times the size of the parent. If the children represent a larger region than the parent then parts of the signal that do not affect the parent can affect the children. This leads to reduced possibilities of finding zerotrees

Following [41], the problem of in-coherent parent-children mappings is solved by us-

ing a limited selection of admissible bases for the WP-decomposition. The limitations are implemented with two constraints on the algorithm for wavelet packet basis search.

The problem presented in Figure 2.12 arise from the fact that a subband that should be a child subband is decomposed one level more than its parent-subband. The constraint for avoiding these situations is formulated as:

"Rule B: When pruning parents, their spatially co-located child subbands may need to be pruned also in order to keep the tree structure valid."

Another mismatch in the original 1:4 parent-children mapping occurs if a child subband is not decomposed enough.

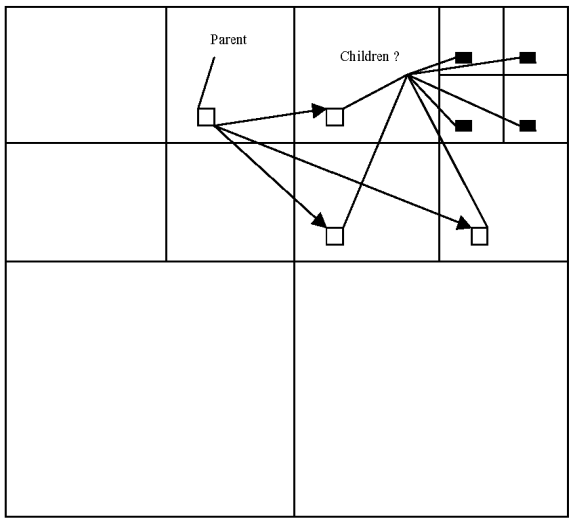


Figure 2.12. A WP-structure that does not have a one to four parent-children mapping.

For the one to four parent-children mapping to work a child subband must be decomposed to the same level as its parent subband or at most one level less. To avoid the problem of child bands not being decomposed enough the constraint added is:

"Rule A: Any four child bands will be considered as candidates for pruning only if their current decomposition level is the same as that of the spatially co-located parent band"

The reordering of a number of child subbands is done so that the resulting structure resembles a subband that is decomposed one level less than the reordered bands. That is, components representing the same spatial region in the child subbands are grouped together. This leads to larger regions representing the same spatial region of the original signal, which should serve as a motivation for reordered subbands to resemble a next higher subband.

Using the decomposition tree we would like to prune nodes in such a way that the tree after pruning is an octave-band decomposition tree of the same depth as the original tree.

The reordering is started from the bottom of the decomposition tree reordering individual components as seen in Figure 2.13. The next level up the reordering has to be made in units of 2x2 components (two-dimensional case) shown in Figure 2.14. This process is continued until the tree shown in Figure 2.15 is achieved.

When the reordering is complete the reordered structure can be fed to the SPIHT coder since now the parent-children relationships are adapted to SPIHT. On the decoding side the bit stream from SPIHT can be decoded producing a reconstruction of the reordered.

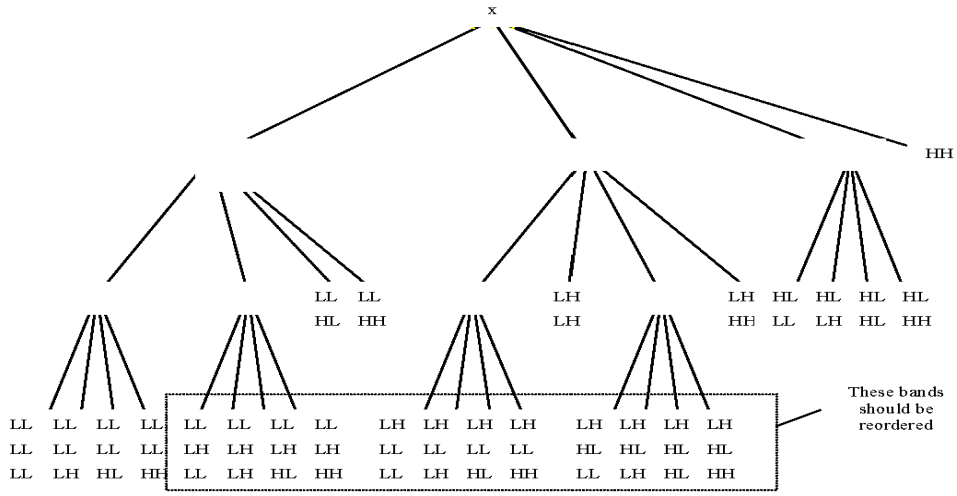


Figure 2.13. The original decomposition structure.

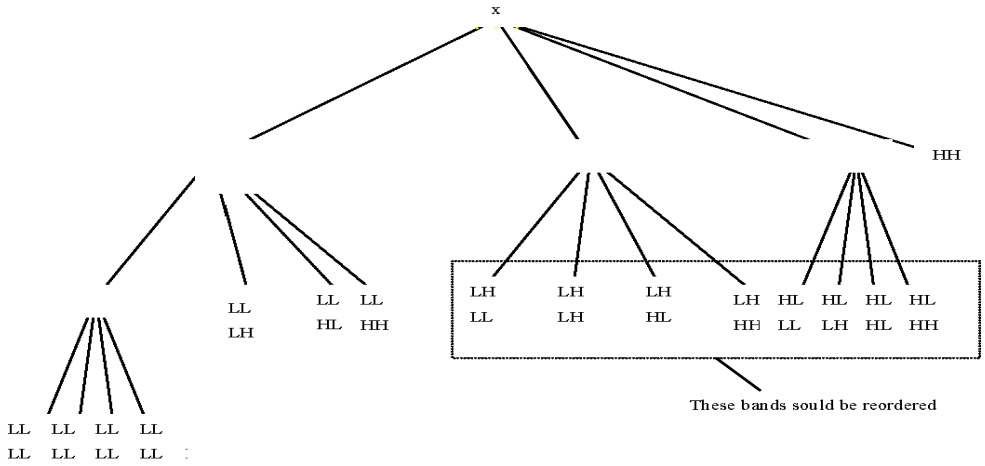


Figure 2.14. The first level of reordering as pruning of the WP tree (the dotted lines mark the spatial location of the subbands that are reordered).

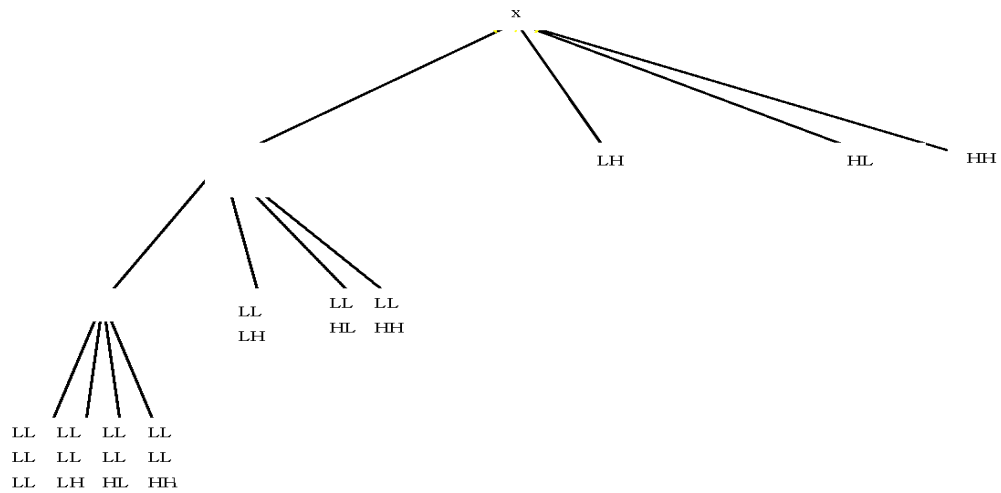


Figure 2.15. Lastly the final stage of reordering, which makes the pruned tree into an octave-band tree.

structure. Before this structure can be inversely transformed an inverse reordering has to be made to transform the structure into its original form. This inverse reordering can also be explained by utilizing the decomposition tree. This time starting from the root each node in an octave-band tree is split (inverse reordered) until the split tree has the same shape as the original tree.

This coder compresses the Barbara image to 32.61 dB at 0.5 bpp.

2.3.5 WP-IMRWD

In [96] an improved version of the coder in section 2.2.5 is applied to a wavelet packet basis. The improvements are the introduction of some new features: 1) Exploitation of the knowledge of component magnitude in the parent band, 2) Reordering of the residual part of the significant map according to the distance to the previously visited point. 3) Employment of hierarchic numerative entropy coder.

As in section 2.2.5 the problem of children nodes having more than one parent must be solved. This is done by applying inverse wavelet packet transform to the parent block until the full resolution parent block is retained. Then forward wavelet packet transform is applied to the retained parent block by using the decomposition basis of the child. Since the layout of the two blocks is now the same, the parent-child relation can be established. The procedure is pictured in Figure 2.16. The performance of the coder is presented in Table 2.7.

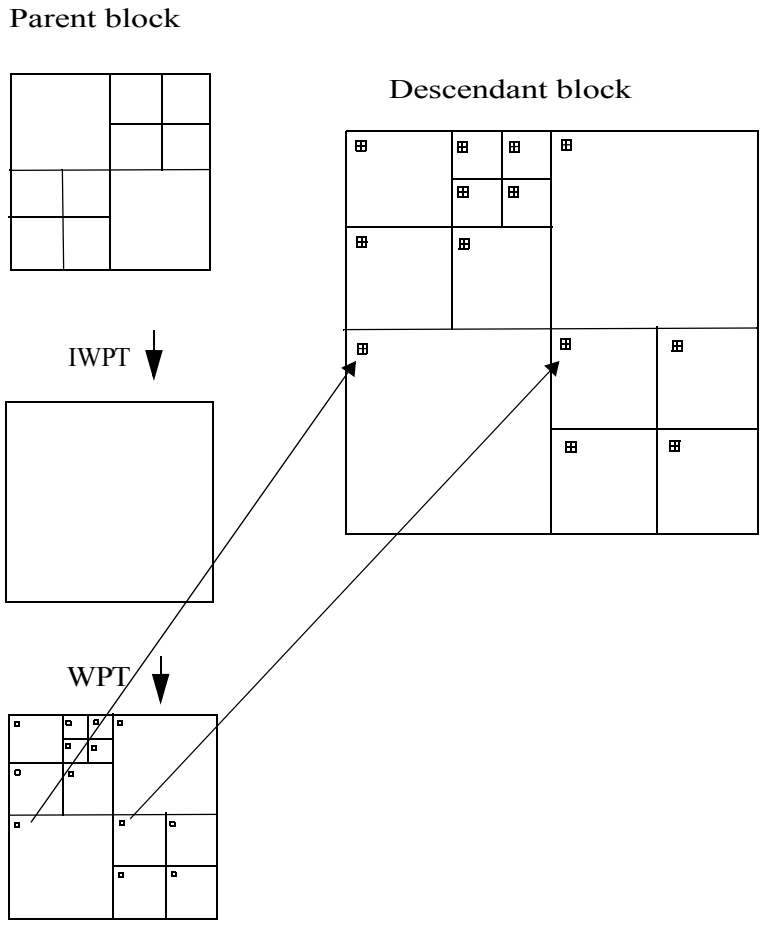


Figure 2.16. Parent-child relation establishing procedure.

	PSNR	
Bitrate	**Barbara**	**Lenna**
1bpp	38.03 dB	40.95dB
0.5bpp	33.05 dB	37.54 dB
0.25bpp	28.52 dB	34.51 dB
0.125 bpp	25.56 dB	31.36 dB

Table 2.7. The performance of the WP-IMRWD coder.

2.3.6 Compatible zerotree quantization of wavelet packets (CZQ-WP)

The issue of wavelet packet zerotrees was also addressed partially by Xiong et al. in the space-frequency quantization (SFQ) algorithm presented in [95], which employs a rate-distortion (R-D) optimization framework to select the best basis and to assign an optimal quantizer to each of the wavelet packet subbands. In their work, however, the subband decomposition was restricted to avoid the *parenting conflict*.

In [68] a general zerotree structure for an arbitrary wavelet packet geometry in an image coding framework is presented. A Markov chain-based cost estimate of encoding the image is used. This adaptive wavelet zerotree image coder has a relatively low computational complexity, performs comparably to the state-of-the-art image coders, and is capable of progressively encoding images.

In order to resolve the parenting conflict, Rajpoot et al. [68] suggest that the child nodes are moved up in the tree so that they are linked directly to the root node of the primary compatible zerotree associated with the family of subbands. This generates a compatible zerotree structure without restricting the basis selection.

The algorithm uses a set of rules to construct the overall compatible zerotree structure for an arbitrary wavelet packet basis. These are reviewed here. The only assumption made is that the lowest frequency subband of the selected basis is always at the coarsest resolution.

Rule (a) If a node P at a coarser resolution is followed only by a node C at the next finer resolution (as in a wavelet transform), the node P is declared as a parent of C.

Rule (b) If a node P is followed by four nodes C_1 , C_2 , C_3 , and C_4 (at the same resolution), then P is declared to be the parent of all these four nodes.

Rule (c) If four subbands P_1, P_2, P_3, and P_4 at a coarser resolution are followed by four subbands C_1 , C_2 , C_3 , and C_4 at the next finer resolution, then node P_i is declared to be the parent of node C_i (for i=1,2,3,4).

Rule (d) If a node P is at a finer resolution than four of its children, say C_1 , C_2, C_3 , and C_4 , then P is disregarded as being the parent of all these nodes and all of them are moved in the tree under a node at the same or a coarser resolution.

In order to ensure that the selected basis is *zerotree friendly*, [68] use a cost function that can estimate the entropy of *quantized* coefficients belonging to the compatible zerotrees. The cost can be estimated by computing the entropy of a discrete random variable

whose value is drawn from the set of codewords used to encode the significance map. The CZQ-WP coder performs as presented in Table 2.8 with the factorized 9-7 biorthogonal filter.

	PSNR	
Bitrate	**Barbara**	**Lenna**
1bpp	36.15 dB	40.10dB
0.5bpp	31.60 dB	37.55 dB
0.25bpp	28.12 dB	34.56 dB
0.1 bpp	24.27 dB	29.95 dB

Table 2.8. The performance of the CZQ-WP coder.

2.4 Segmentation-based image coding

During the past years there have been many different approaches to achieve better image compression than the JPEG standard. The wavelet transform is now a part of the JPEG2000 standard but the wavelet packet techniques have not yet reached their ultimate application and performance. One important drawback of the methods described so far in this chapter is that they all operate on the entire image. One approach is to make the assumption that the wavelet packet best basis search finds the resonance in the image and thereforee should be applied to pure textures only. This is done by segmenting the image and finding the best basis for each segment.

2.4.1 Edge separating wavelet image coding

Wavelet-based image coders are good at representing smooth regions and isolated point singularities. However, they are less adaptive at representing perceptually important edge singularities, and coding performance suffers significantly as a result. In [28] Froment & Mallat proposed a two-stage image coder framework based on modelling images as edge cartoons plus textures. The coder works by first inferring and efficiently coding the edge information from the image using a multiscale wavelet decomposition. Second, the residual, “edgeless” texture image is coded using a standard wavelet coder. This preliminary coder improves significantly over standard wavelet coding techniques in terms of visual quality, according to [28].

2.4.2 Joint use of segmentation and wavelet packet

The best way to find an optimal basis representation for an image is to search jointly for the best segmentation and wavelet packet tree. This is the idea behind the following methods.

Double tree (DT)

The wavelet packet tree adapts to frequency content, the spatial quadtree adapts to spatial content. The spatial quad tree does not have the same capacity as a WP tree for compacting the energy of the image. The double tree was first suggested by Herley et al. [32,33] to address the problem of non-stationarity without sacrificing the energy compaction, by which is meant the deterministic non-stationarity described in section 1.1.4. The expansion finds the optimal wavelet packet trees over spatial segments of the image. This algorithm is extended to the use on images in [34]. The algorithm calculates the best wavelet packet tree for the entire image and stores the cost. Then the image is segmented into a first level quad tree and separate wavelet packet trees are calculated for each of the four segments and the cost is stored. Then the image is segmented into a second-level quad tree and separate wavelet packet trees are calculated for each of the sixteen segments and the cost is stored. This is iterated until sufficient depth of the quad tree is reached. The overall basis search identifies the most efficient dyadic segmentation and corresponding wavelet packet expansions for the segments.

The double tree performance with a 7-9 tap linear phase filter and uniform quantization is shown in Table 2.9 [34].

	PSNR	
Bitrate	**Barbara**	**Lenna**
1bpp	36.55 dB	39.32 dB
0.5bpp	31.73 dB	36.26 dB
0.25bpp	28.06 dB	33.24 dB

Table 2.9. Performance of the double tree coder.

The dual double-tree (DDT)

The dual double-tree in Smith & Chang [82] finds the optimal spatial segmentations of the wavelet packet nodes. The dual double tree is produced by first growing a single wavelet packet tree followed by the full segmentation of all subbands. The dual double tree performance with a 7-9 tap linear phase filter and uniform quantization is shown in Table 2.10 [34].

	PSNR	
Bitrate	**Barbara**	**Lenna**
1bpp	36.71dB	39.72 dB
0.5bpp	32.03 dB	36.62 dB
0.25bpp	28.27 dB	33.49 dB

Table 2.10. Performance of the dual double tree coder.

The joint space and frequency best basis

Both the double tree and the dual double tree provide an asymmetric treatment of space

and frequency. As a result, neither tree attains the full joint decomposition in space and frequency.

An algorithm that considers both frequency and spatial split at each node is presented in [82]. It combines the double-tree and dual double-tree to attain the best joint space and frequency decomposition for image compression.

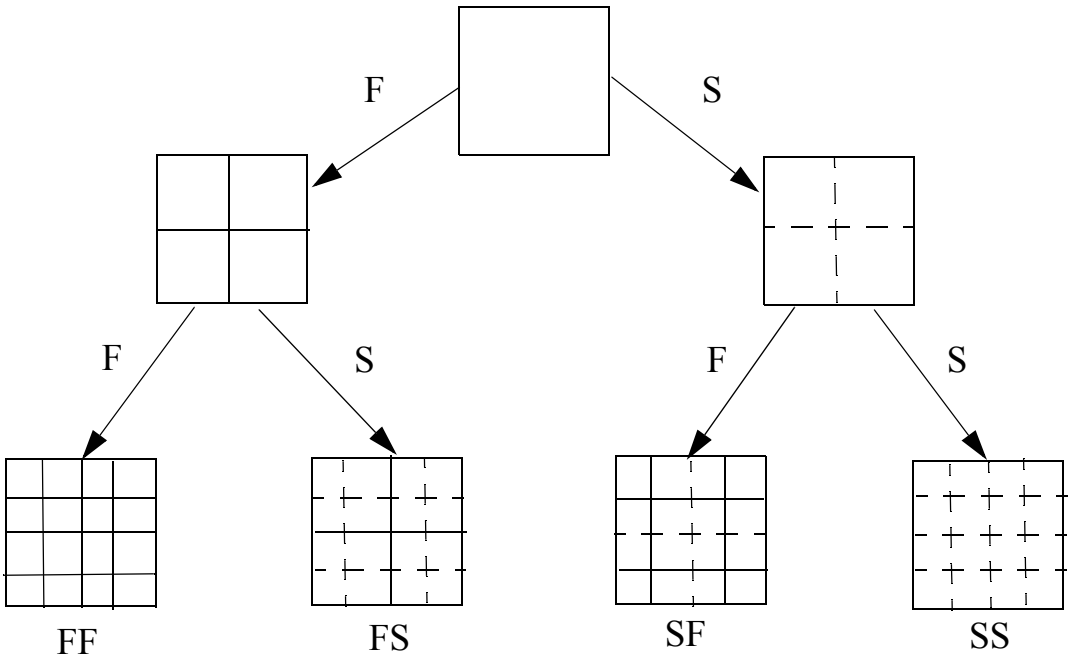


Figure 2.17. The best joint space and frequency decomposition uses both frequency and spatial split at each node. Solid lines describe frequency split F, dashed line describe spatial split S.

In Figure 2.17 the two step joint space and frequency decomposition is shown. Solid lines describe frequency split F, dashed lines describe spatial split S. If we approximate the output from FS to be equal to the output from SF these data need not be produced twice. The accuracy of this approximation is filter dependent. The decomposing can be viewed as the graph in Figure 2.18.

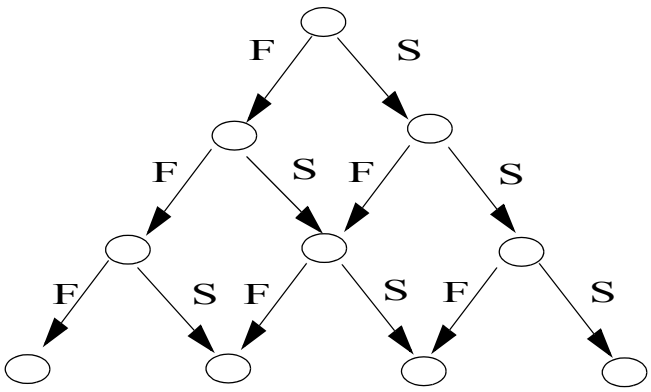


Figure 2.18. Joint spatial frequency decomposition graph.

The left branch FFF provides the fully decomposed wavelet packet tree while the right branch SSS provides the fully decomposed spatial quadtree. The middle nodes provide different combinations of space and frequency decompositions where DT is one and DDT is another. This gives a larger set of possible bases to choose from when selecting the best basis. An example for a one-dimensional (time-varying) signal is shown in Figure 2.19 [82].

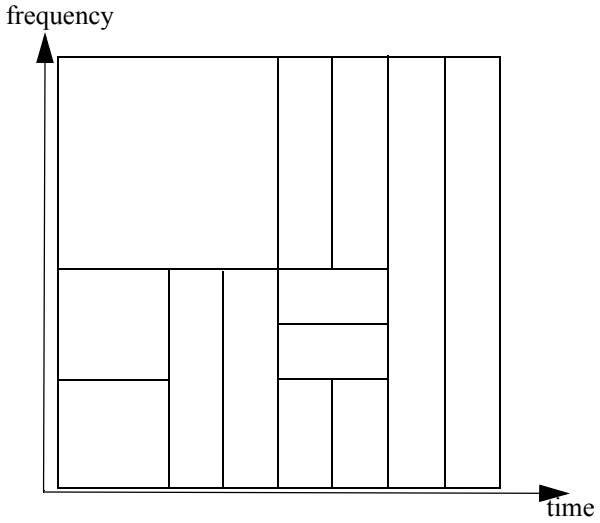


Figure 2.19. One example of a time frequency tiling possible with the basis set in Figure 2.18 but not with WP, DT or DDT.

Performance of the joint space and frequency decomposition of the Barbara image is shown in Table 2.11, when using a R-D optimized cost function for best basis selection [82].

	PSNR	
Bitrate	**Haar filter**	**QMF filter**
2.0bpp	43.8 dB	43.8dB
1.0bpp	37.7 dB	37.5 dB
0.5bpp	33.0 dB	22.7 dB

Table 2.11. Performance of the joint space and frequency decomposition of the Barbara image using different filters.

2.5 Summary

In this chapter we have presented some image coding algorithms using wavelet and wavelet packet methods that have contributed to the development of the field. The wavelet compression methods have found their way into use in standards while the wavelet packet algorithms still struggle with problems.

The algorithms presented in this chapter are typical in the sense that they focus on the quantization, bit allocation, and entropy coding steps of the coding chain shown in Figure 1.10. The problem of choosing the basis is often not treated apart from the basic scheme of each method. In the wavelet case one can argue that the basis is fixed but the result still changes with the choice of filters. For the wavelet packet case the bases are totally different depending on the image, the choice of filter, and the choice of cost function. Also the choice of basis selection method, bottom-up or a greedy top-down search affects the result. This is not treated in the earlier works.

The wavelet packet methods demonstrate the difficulties of using the principle to get a good representation of the image for compression purpose. The main problem identified is the use of wavelet packets on the entire image while an optimal basis is only found for pure textures. Different filters are also shown to influence the results. But the wavelet packet methods also demonstrate the potential to find a good representation, although the problems are not yet solved.

In the next chapter we will investigate the wavelet packet trees in more detail and how the different bases depend on image structure and on the choice of filter as well as cost function.

Chapter 3

The Wavelet Packet Triplet Problem

When working with image coding algorithms using the wavelet packet transform different bases appear as the best basis for the same image when using different cost functions or filters. For different images, the best compression is achieved with different bases due to the different image statistics. This is almost intuitively understood. The triplet of image, filter, and cost function is closely related in the analysis of the compression ability of the wavelet packet transform. The influence of filter and cost function in connection with different image statistics is analysed in this chapter.

We start the presentation of the triplet problem by analysing a database of coding results using different filters and cost functions on a subset of the Brodatz texture images. The data base contains coding results from 2295 different triplets, the combination of 17 images, 15 filters, and 9 cost functions. The database is analysed and discussed with respect to images, cost function effects, and filter effects. Due to the limited space in this thesis only a part of the result is presented in plots. From this we get some guidance how to choose filter and cost function in the implementation of one of the wavelet packet coders from Chapter 2. The same filter and cost function is also used in the next part of the chapter where we present a method to analytically calculate the expected wavelet packet basis, with a top-down approach. Here we use a signal model to represent the image. We then apply this method to different images and compare the top-down calculated bases with the bottom-up best bases for the same images.

We use the expression *best basis* for the basis found by the best basis search algorithm of Coifman & Wickerhauser [17]. This algorithm uses a bottom up search strategy and is stated to find a global optimal basis. The expression *wavelet packet basis* is used for bases found to be optimal by any search algorithm. In this part of the thesis we are mainly concerned with the compression ratio achieved by the wavelet packet transform, and restrict ourselves to estimating the entropy of the components as a measure of possible compression performance except in one section where we test the performance of the SPIHT coder and the CWP coder.

3.1 Decomposing the Brodatz textures

The reason for using texture images in this work is that we are specifically interested in efficient representation of the resonance in the image. We have used the Brodatz collection of texture images for most of our experiments [10]. The test images available for our experiments are presented in Appendix A, and are a set of 17 digitized images out of the collection in [10].

The images have different resonance due to their different textures. The cost function applies to the output from the filtering process. This leads to the questions: Does the filter matter? What is the result if another cost function is used? The resulting best bases when applying the Haar filter and Db4 filter with the *energy entropy* cost function, specified in Equation 3.12, and the *log energy* cost function, specified in Equation 3.14, respectively to the images in Figure 3.1 are shown in Figure 3.2 - 3.7.

Clearly we can see that the change of filter or cost function results in different bases. The different bases representing the best basis for the triplet *image, cost function* and *filter*, raise several questions. Why is there no typical tree for a particular image or why is there no strong trend for a certain type of tree in combination with a fixed filter? Which combination is best for compression of the image?

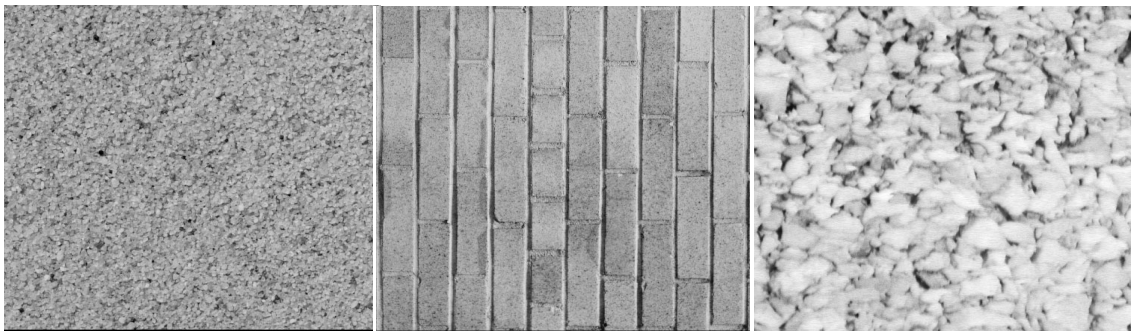


Figure 3.1. The textures analysed in Figure 3.2 - 3.7, beach, brickwall, gravel.

3.2 Results from the triplet database

In this section the compression performance of the wavelet packet transform is evaluated through extensive testing of all combinations of 17 images, 9 cost functions and 15 different wavelet filters. The motivation is to compare the filter theory and cost function discussion with experimental results and find explanation as to why the common adhoc use of wavelet packets does not always work as expected.

3.2.1 Entropy estimations

An important measure in image coding applications is the cost for coding the transform components. It may be even more important than the cost for choosing and representing the basis. We simulate the coding cost by measuring the entropy of the components by the theoretically calculated entropy presented in Equation 3.1.

The image is represented by a sequence of pixels, $\bar{x} = (x_1, x_2, \ldots, x_{N\times N})$, created by a scan of the two-dimensional image array, where x_i is the pixel at location i, $n = N \times N$ is the number of pixels in the image, as a message generated by a discrete source together with a set of probabilities, $p(x_i)$. Then the image is a finite dimensional random vector variable $X = (X_1, X_2, \ldots, X_n)$ where X_i is the pixel random variable at location i. In limes, $n \to \infty$, the random vector is a random process, and the random process is the infinite dimensional generalization of a vector random variable [22]. One image is a realisation of the random process. We assume ergodicity and thereforee we can use one realisation of our process (one image) to compute statistics [74].

Let p_{xk} be the probability that the pixel x takes a value k out of an alphabet defined by the set of amplitude values, then the entropy for this pixel x is

$$H(x) = -\sum_{k=1}^{N} p_{xk} \log p_{xk} \tag{3.1}$$

and we know that we can represent the pixel with H bits/ pixel. We want to know the entropy of the whole image or of the part of the image representing the subband. When we consider two pixels at the same time we use the joint probability $p(i,j)$ for the pixels x_i and x_j, $p(i,j)$ can be represented by the relative frequency of the pair *(i,j)*, all *i,j*.

$p(i,j) = p(i)(p(j))$ only if x_i and x_j are independent. The entropy for the pair of pixels is the joint entropy $H(x_i, x_j)$ and we have

$$H(x_i, x_j) \le H(x_i) + H(x_j) \tag{3.2}$$

with equality if and only if the pixels x_i and x_j are independent!

For the whole image we can write the joint entropy as

$$H(\text{image}) \le \sum_{i=1}^{n} H(x_i) \tag{3.3}$$

and if the pixels values are drawn from the same distribution then

$$H(\text{image}) \le nH(x_i) \tag{3.4}$$

with equality if they are independent.

The entropy rate of the random vector variable $\bar{X}$ representing our image is

$$\frac{H(\text{image})}{n} \le H(x_i) \quad \text{bits / pixel.} \tag{3.5}$$

The collection of random variables in a vector is independent if and only if the joint (for the whole image) probability density function (pdf) $p(x_1, x_2, \ldots, x_{N\times N})$ can be written as

$$p(x_1, x_2, \ldots, x_{N\times N}) = p(x_1)p(x_2)\ldots p(x_{N\times N}) \tag{3.6}$$

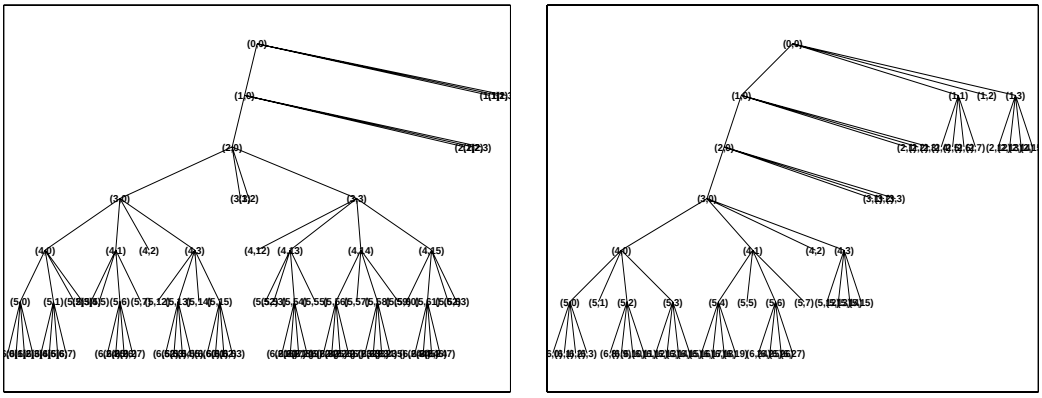


Figure 3.2. The beach texture, its best basis tree using Haar filter and the ***energy entropy*** cost function to the left and Haar filter and the ***log energy*** cost function to the right.

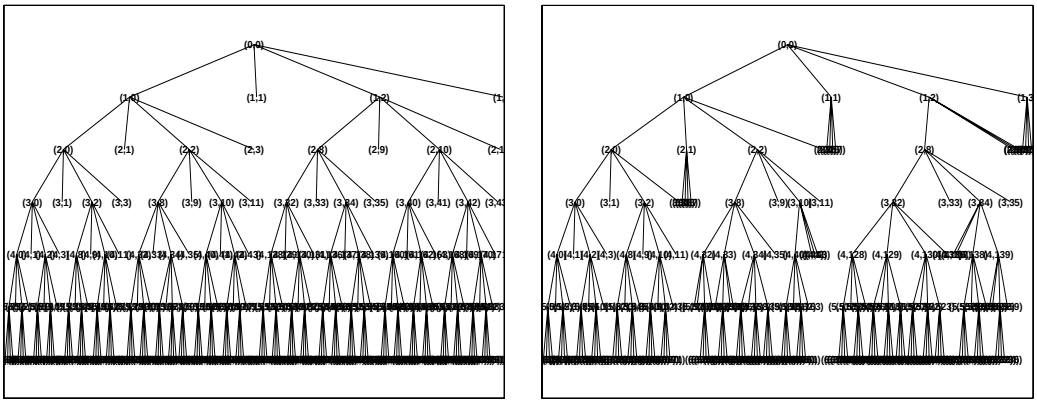


Figure 3.3. The brickwall texture, its best basis tree using Haar filter and the ***energy entropy*** cost function to the left and Haar filter and the ***log energy*** cost function to the right.

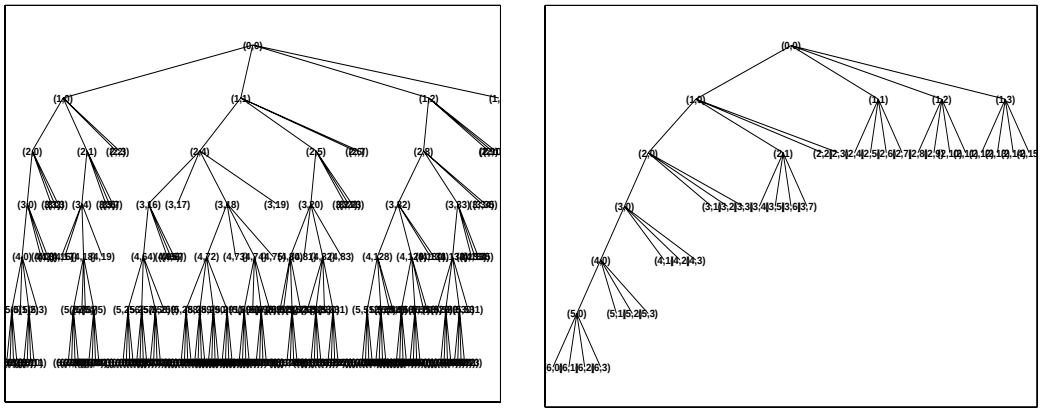


Figure 3.4. The gravel texture, its best basis tree using Haar filter and the ***energy entropy*** cost function to the left and Haar filter and the ***log energy*** cost function to the right.

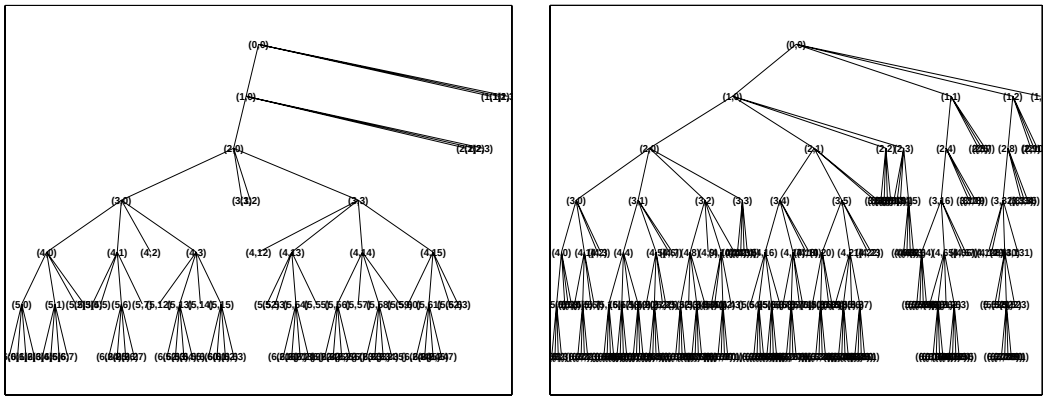


Figure 3.5. The beach image, its best basis tree using the Haar filter and the ***energy entropy*** cost function to the left and the DB4 filter with the ***energy entropy*** cost function to the right.

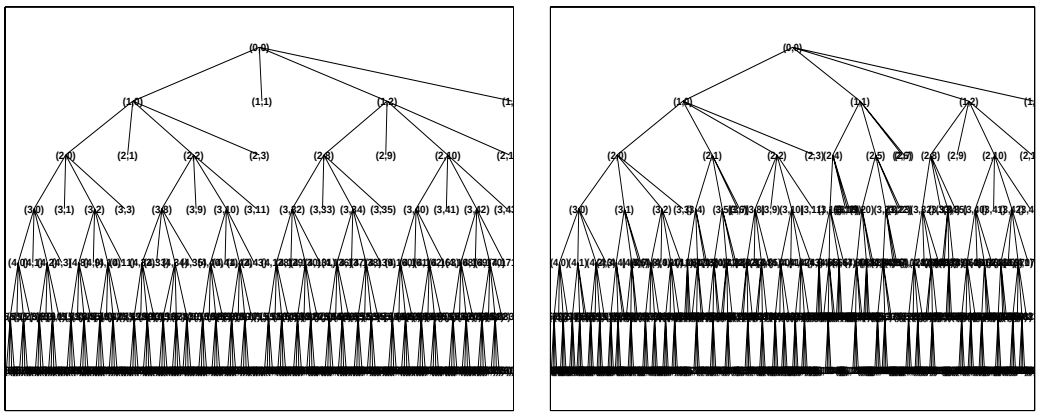


Figure 3.6. The brickwall image, its best basis tree using the Haar filter and the ***energy entropy*** cost function to the left and the DB4 filter with the ***energy entropy*** cost function to the right.

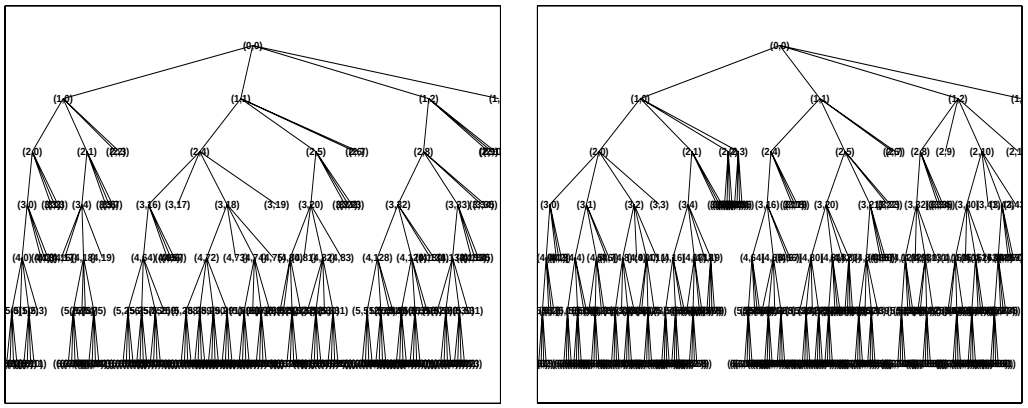


Figure 3.7. The gravel image, its best basis tree using the Haar filter and the ***energy entropy*** cost function to the left and the DB4 filter with the ***energy entropy*** cost function to the right.

If each random variable has the same distribution we have an iid (independent, identically distributed) random vector X.

The probability density function (pdf) of a pixel describes how probable a pixel value is. The relative frequency of the pixel value measures the probability of the pixel value independent of its neighbours. This memory-less assumption, together with the ergodicity assumption, is the common approximation made when estimating the pdf with the grayscale histogram of the image.

Common models for the pdf of a memory-less source, used in image processing, are the Laplacian density

$$p(x) = \frac{1}{\sigma_x\sqrt{2}} e^{-\frac{\sqrt{2}|x|}{\sigma_x}} \quad , m_x=0, \tag{3.7}$$

and the Gaussian density

$$p(x) = \frac{1}{\sigma_x\sqrt{2\pi}} e^{-\frac{(x-m_x)^2}{2\sigma_x^2}} \tag{3.8}$$

The generalized Gaussian density

$$p(x) = Ke^{-(|x|/\alpha)^\beta} \tag{3.9}$$

$$K = \frac{\beta}{2\alpha\Gamma\left(\frac{1}{\beta}\right)} \qquad \Gamma(t) = \int_0^\infty e^{-u}u^{t-1}du \tag{3.10}$$

can be used as a model of the pdf [54,23]. α models the variance and β is inverse proportional to the decreasing rate of the peak. This model can represent both the Laplacian (β=1), and the Gaussian (β=2) distribution as special cases.

For a memory-less source we can estimate the pdf with the grayscale histogram of the image and the entropy rate of the source is given by Equation 3.1.

The entropy of the pixel value is the entropy true to the intentions of Shannon and common in the telecommunication community [77]. The calculated entropy of the pixel value gives us a hint of the compression rate achieved for a memory-less source. In order to estimate the coding performance, the entropy of the pixel value has been used in our experiments knowing that we can have better compression with source coding algorithms that take care of the memory in the signal. The cost of representing the basis tree structure is not taken into consideration here. This is because we are interested in the properties of the transform alone and the cost of coding the decomposition tree is minor compared with the cost of coding the values and positions of the transform components. Fine quantization of the components is used as these are typically represented with 8 bpp. The transform itself generates lossless compression with perfect reconstruction filters. No efficient

bit allocating procedures are here applied on the component matrix in order to achieve better compression.

3.2.2 Cost function effects

Here we describe the cost functions used in the triplet database. The behaviour of the cost functions is very different depending on the range of the value of the components. The norm |x| and x^2 have the same basic behaviour in the interval 0..1 as in 1..255 but cost functions based on log(x) show different characteristics in the two different intervals. Although the curve form of the functions $\log(x^2)$ and log(x) looks the same in the two intervals, working in one interval gives large negative values for inputs close to the lower end of the interval and values close to zero for the upper end of the interval while the other interval gives values close to zero for inputs close to the lower end of the interval and large values for the upper end of the interval

The functions $x^2\log(x^2)$, and xlog(x) show an even larger difference in the two intervals, see Figure 3.8 and Figure 3.9. In the interval 0..1 these functions are close to zero at both ends of the interval with a maximum negative value in the middle of the interval while in the interval 1..255 the functions are dominated by their x and x^2 components and can almost be approximated by them.

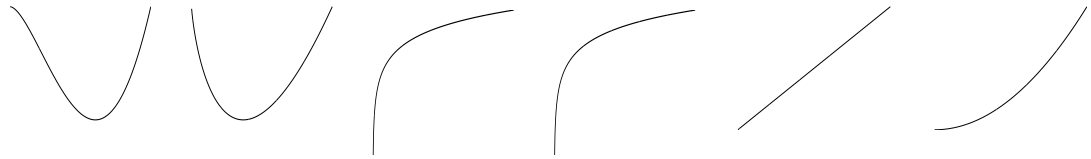


Figure 3.8. For the interval 0..1, $x^2\log(x^2)$, xlog(x), $\log(x^2)$, log(x), x, x^2.

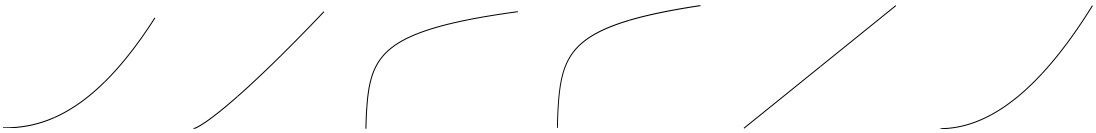


Figure 3.9. For the interval 1..255, $x^2\log(x^2)$, xlog(x), $\log(x^2)$, log(x), x, x^2.

In the following, the inputs are in the interval 1..255. This is why the energy entropy shows almost the same results as the quadratic norm in most of the plots. Cost functions based on the form $x^2\log(x^2)$ and xlog(x) were originally designed for use on the probability distribution, on the interval 0..1 [77] and have the originally intended properties only in images normalized to the interval 0...1.

The best basis algorithm is used in the construction of the triplet database. The algorithm is dependent on the assumption that the cost function is additive. An additive cost measure is defined by

$$M = \sum_i C(x_i) \tag{3.11}$$

where $C(x_i)$ is the cost for the signal component *i*.

The different cost functions we have used are presented below.

Entropy of the pixel energy

$$C_{sh} = -\sum_k |x_k|^2 \log |x_k|^2 \tag{3.12}$$

This is the cost function used in the wavelet packet community under the misleading name Shannon entropy. In this thesis we call this cost function the *Energy Entropy*. This cost function is applied to the subbands components in the interval 1..255, thus it functions almost like the quadratic norm.

We estimate the true entropy of a subband by calculating

Entropy of a subband

$$C_{hest} = -\sum_{x=1}^{N} p(x) \log p(x) \tag{3.13}$$

with $p(x)$ being the normalized histogram of the same subband. This is a fair approximation of the entropy of the subband as long as the subband is not too small. It should be noted that this measure, if used as a cost function, is not additive.

Logarithmic cost

$$C_{\log} = \sum_k \log |x_k|^2 \tag{3.14}$$

The logarithm of the pixel energy is used as a cost function applied on the image in the interval 1..255.

The L^p -norm

$$C_{np} = \sum_k |x_k|^p \tag{3.15}$$

The L^1-norm and the $L^{1.99}$-norm have been used as the cost function most likely to match the Laplacian and the Gaussian probability distributions. The L^1 - norm, representing the absolute mean of the grayscale values, and the L^2 - norm are popular measures. The energy measure L^2 - norm, the quadratic norm, representing the variance of the grayscale values if the mean = 0, is not useful to make tree-pruning decisions when the decomposition is orthonormal. We use the $L^{1.99}$ - norm as a measure close to the energy measure and we use some norms lying in between to see if any of them possibly match some generalized Gaussian distribution.

3.2.3 Measurements of cost function effects

We first present results on how cost functions affect the decompositions. We compare the entropy estimations for best bases chosen with different cost functions, while the other two variables are held constant.

Consider the texture images in Figure 3.10, brickwall and cslfleath.

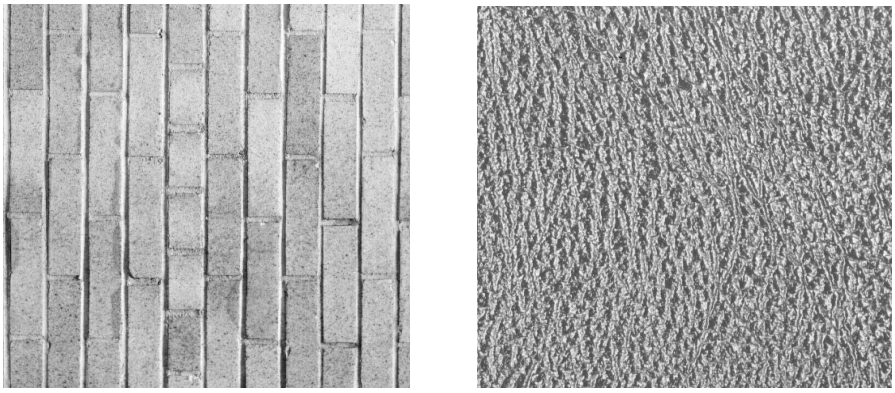


Figure 3.10. The images brickwall (left), cslfleath (right).

In Figure 3.11, and Figure 3.14 the images are analysed with the Haar filter and their associated best bases are chosen with the different cost functions described in section 3.2.2. The cost of coding the components, estimated by Equation 3.1, from the best basis is marked by o and the cost of coding the components from the corresponding wavelet basis is marked by x. It is expected that the x:s form a horizontal line as the measure is taken on the same wavelet tree using the same decomposition filter. Analysis of more images is collected in Appendix B.

For both images, the wavelet packet basis performs better than the wavelet basis. Figure 3.12 and Figure 3.13 show the bases chosen with the Haar filter and the cost functions defined in Equation 3.13 and Equation 3.14 for the image *cslfleath* in Figure 3.10. These figures show basis trees with very different structure, and performance worse than the other cost functions, but still better than for the wavelet basis. However, note the scaling of the y-axis, in bits/pixels. The difference is very small! Since we assume lossless coding without coarse quantization of the transform components the visual performance (measured in PSNR) is not affected, hence the performance of different bases is almost in-distinguishable.

The next example image (brickwall) produces similar results with the analysis filter Haar. Figure 3.15, Figure 3.16 and Figure 3.17 show bases with different performance. The basis in Figure 3.15 looks similar to the wavelet basis. However, this is not the basis with performance most similar to the wavelet basis; that goes to the basis in Figure 3.16, with performance in position 3 in Figure 3.14.

While the bases chosen with the Haar filter and different cost functions all perform very close to the wavelet basis performance, we see another result when choosing another analysis filter. When the images are analysed with the bior1,5 filter we see that instead of having the difference in performance of different cost functions in the range of 0,01 bit/pixel we now have the difference in 2 or 3 bit/pixel between best and worse performance, Figure 3.18.

The plots in Figure 3.18, Figure 3.19, and Figure 3.20 representing the analysis with the bior1.5 filter and the db4 filter clearly demonstrate the importance of using a filter and a cost function that work well together. For some combinations of filter, cost functions and image, the best basis selection algorithm does not work well at all. Either the fully decomposed tree is chosen or the algorithm chooses not to decompose the image at all, giving the entropy of the image itself. The fully decomposed image has reduced entropy but the best result is achieved with a carefully chosen best basis that represents the resonance in the image.

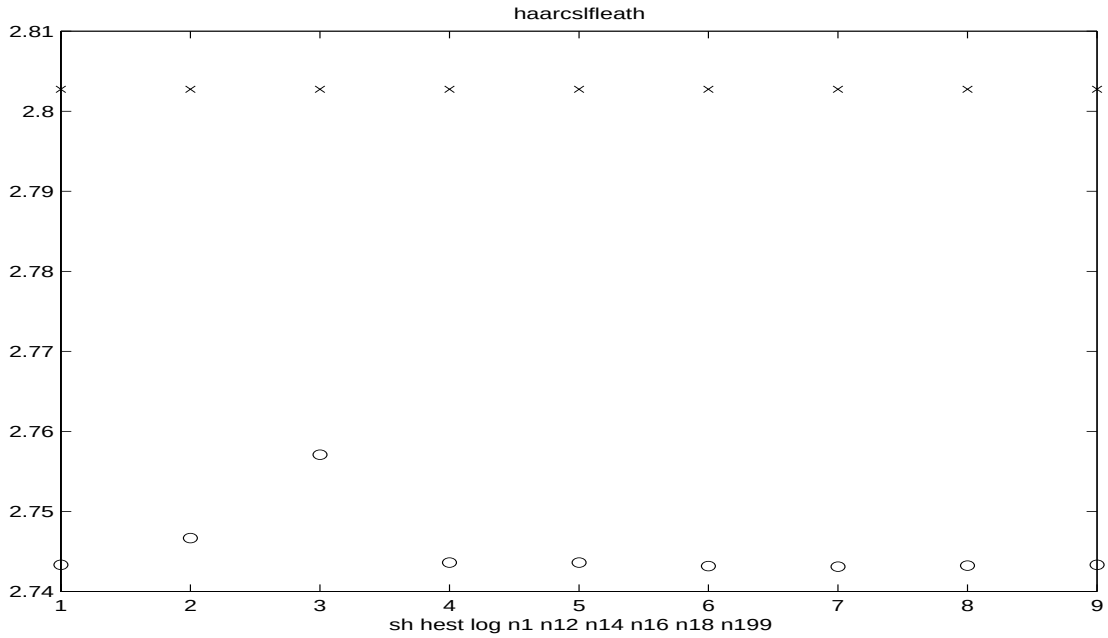


Figure 3.11. The cslfleath image decomposed with the Haar filter and where the cost function is varied. The cost of coding the components of the best basis is marked by **o** and the cost of coding the components of the corresponding wavelet basis is marked by **x**.

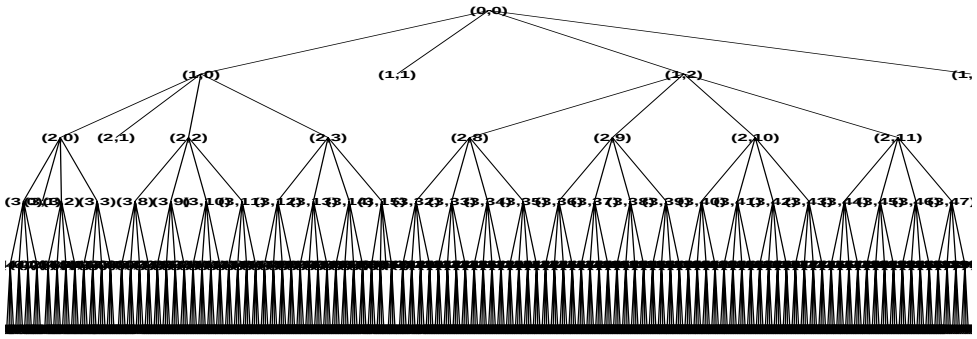


Figure 3.12. The best basis for the triplet filter Haar, image cslfleath, and cost function hest, defined in Equation 3.13. The estimated entropy is shown in position 2 in Figure 3.11.

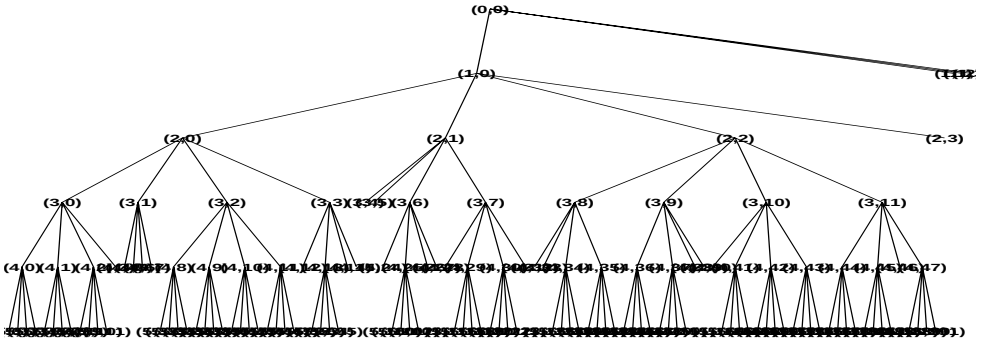


Figure 3.13. The best basis for the triplet filter Haar, image cslfleath, and cost function log, defined in Equation 3.14. The estimated entropy is shown in position 3 in Figure 3.11.

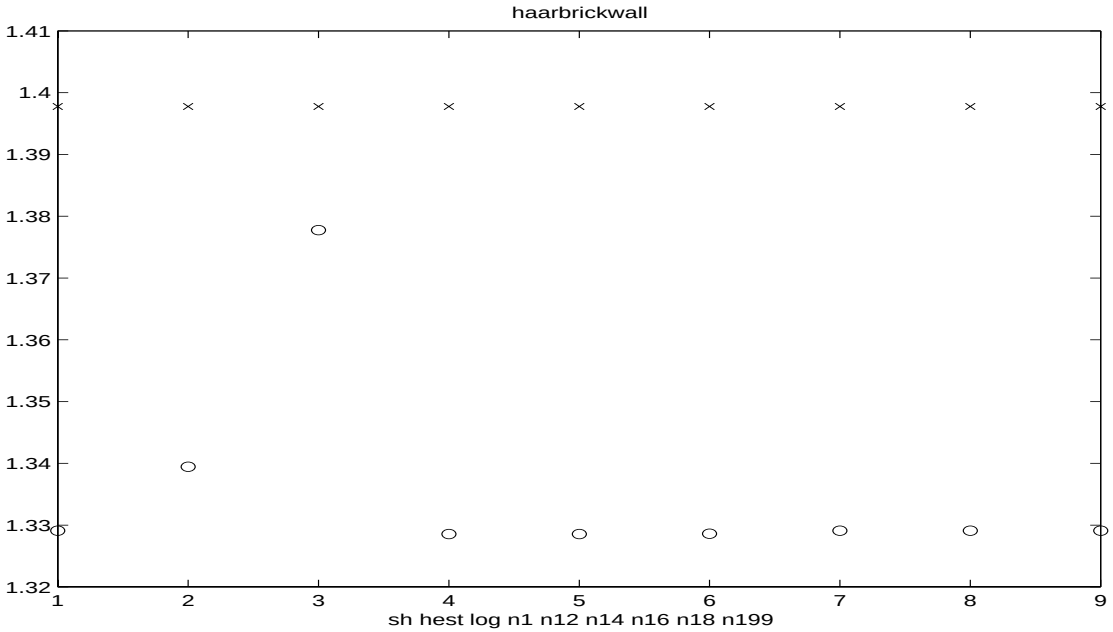


Figure 3.14. The brickwall image decomposed with the Haar filter and where the cost function is varied. The cost of coding the components of the best basis is marked by **o** and the cost of coding the components of the corresponding wavelet basis is marked by **x.**

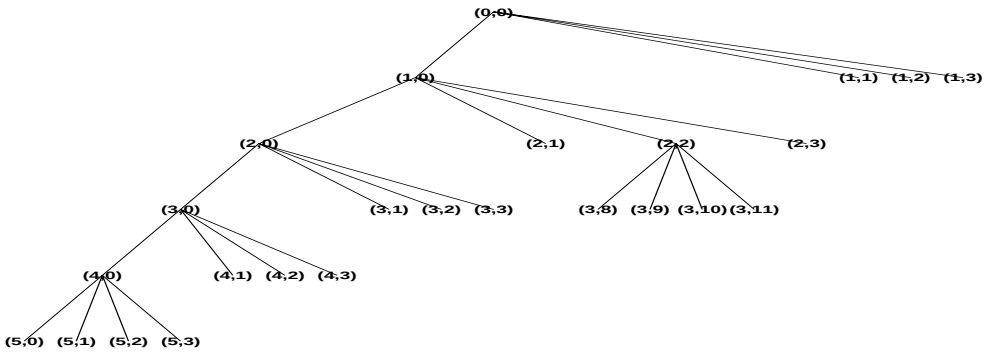


Figure 3.15. The best basis for the triplet filter Haar, image brickwall, and cost function hest, defined in Equation 3.13. The estimated entropy is shown in position 2 in Figure 3.14.

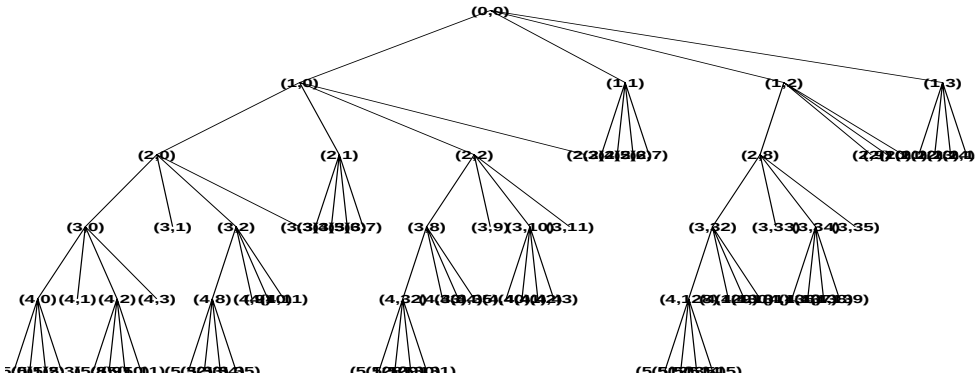


Figure 3.16. The best basis for the triplet filter Haar, image brickwall, and cost function log, defined in Equation 3.14. The estimated entropy is shown in position 3 in Figure 3.14.

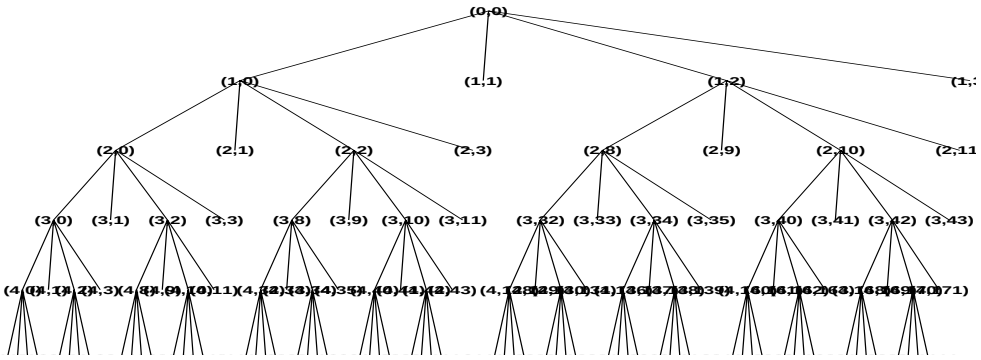


Figure 3.17. The best basis for the triplet filter Haar, image brickwall, and cost function sh, defined in Equation 3.12. The estimated entropy is shown in position 1 in Figure 3.14.

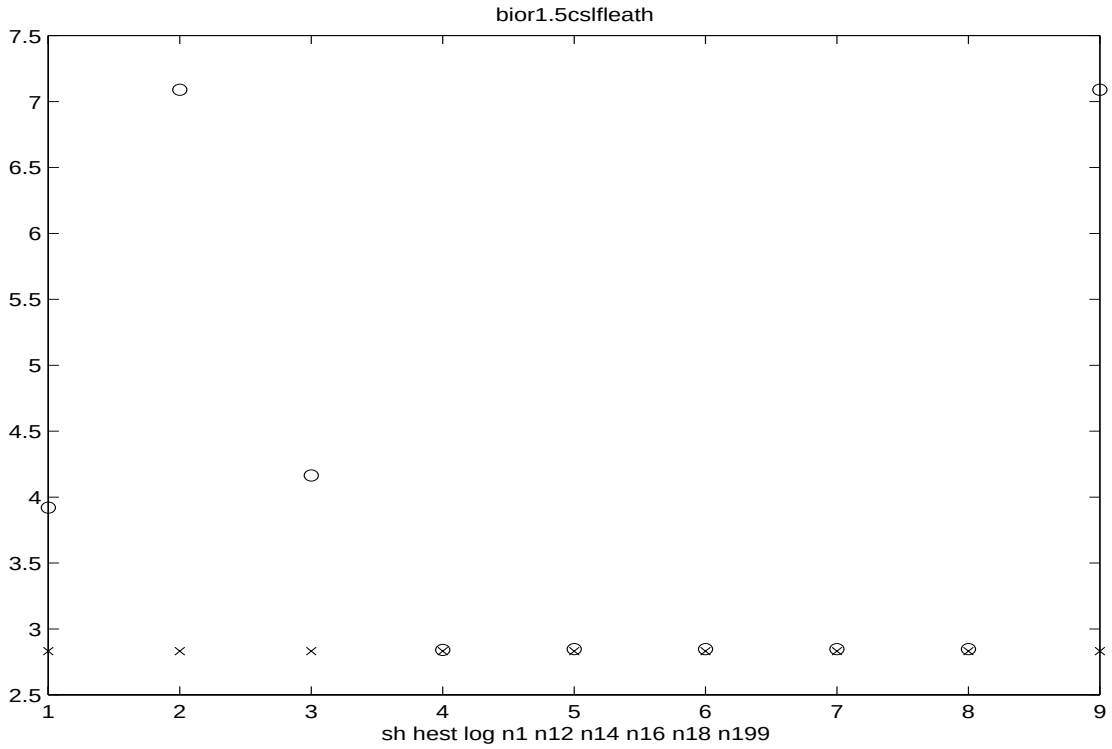


Figure 3.18. The cslfleath image decomposed with the filter bior1.5 and where the cost function is varied. The cost of coding the components of the optimal wavelet packet basis is marked by **o** and the cost of coding the components of the corresponding wavelet basis is marked by **x.**

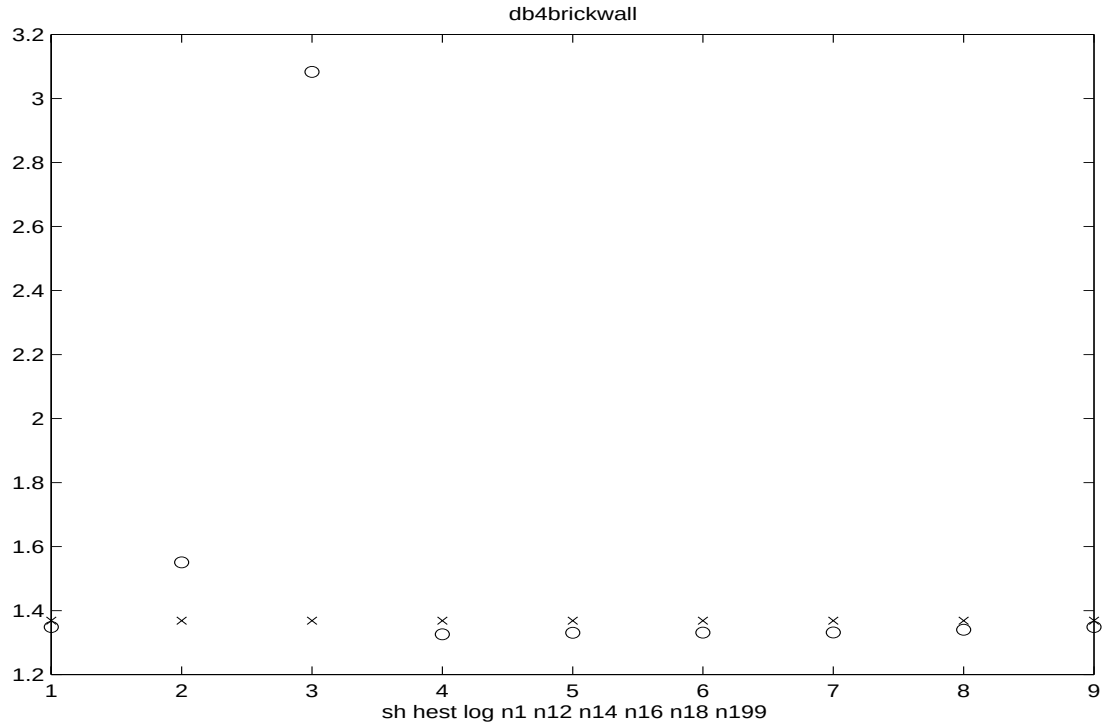


Figure 3.19. The brickwall image decomposed with the db4 filter and where the cost function is varied. The cost of coding the components of the optimal wavelet packet basis is marked by **o** and the cost of coding the components of the corresponding wavelet basis is marked by **x**.

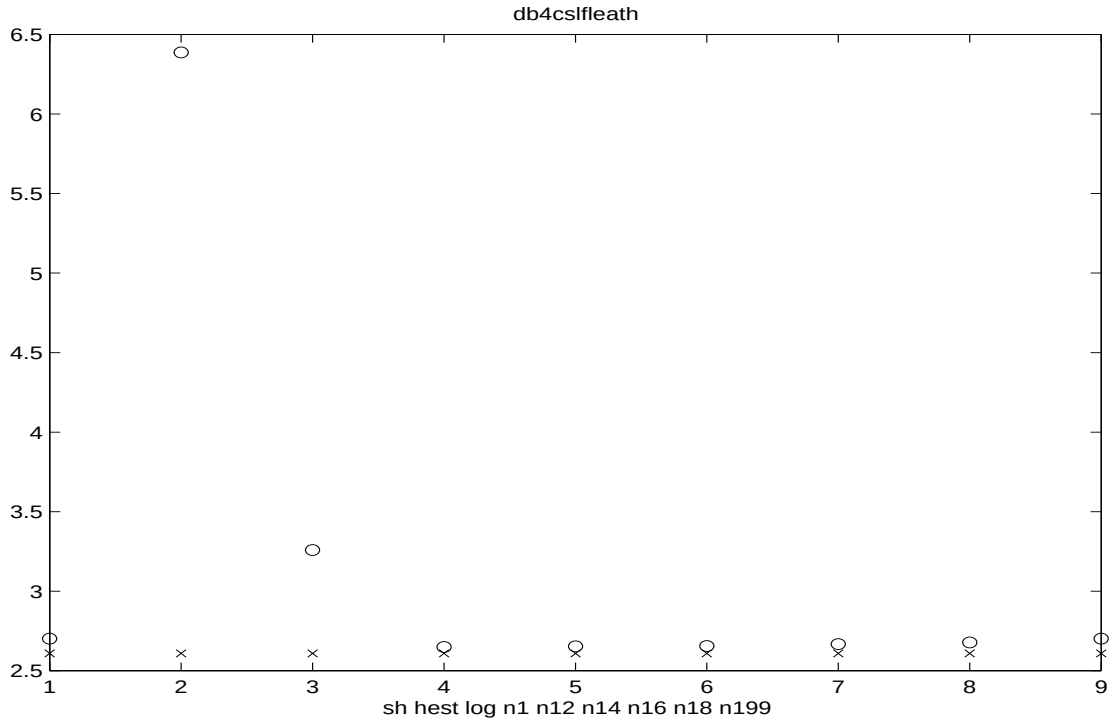


Figure 3.20. The cslfleath image decomposed with the db4 filter and where the cost function is varied. The cost of coding the components of the optimal wavelet packet basis is marked by **o** and the cost of coding the components of the corresponding wavelet basis is marked by **x**.

3.2.4 Filter effects

In this section we review some filter properties important to image analysis and compression and present the filters used in the triplet database. This is done in preparation for the analysis of the triplet database from a filter point of view presented in section 3.2.5. The results presented in this section are proved in the text books [19,56,90,93] if no other reference is given.

Orthogonal filters implement a unitary transform between the original and its subbands. One of the important features of unitary transforms is conservation of energy. This is essential in section 3.4. For compression applications we want the wavelet and wavelet packet transform to produce only a few non-zero components.

Definition. *Vanishing moments* [56]. Ψ has p vanishing moments if

$$\int_{-\infty}^{\infty} y^k \psi(t)dt = 0 \qquad 0 \le k < p \tag{3.16}$$

Suppose we want to decompose a function $F(x)$ into wavelets. We compute all the wavelet components $\langle F, \psi_{j,k} \rangle$. If F is L-1 times continuously differentiable and Ψ has L *vanishing moments*, we will only have large values of the wavelet components for singularities of F or its derivatives. This is why the edges are clearly visible in the transform images. This also means that the signal properties dictate the number of vanishing moments required for the wavelet to be able to decompose the signal in a manner suitable for compression.

Because of the *cascading effects* of the wavelets in a wavelet or wavelet packet decomposition we get accumulated errors from the filtering process. The decomposition can be described by the equivalent filter $H_r(z)$ obtained by the cascading of the filters H_0 and H_1.

$$H_r(z) = \prod_{k=1}^{d_r - 1} H_{P_r}(z^{2^k}) \tag{3.17}$$

where d_r is the depth of the cascade of path r and $P_r(k) \in \{0, 1\}$ indicates which of the filters H_0 or H_1 is used.

Linear phase is desired since such filters can be cascaded in a pyramidal filter structure without the need for phase compensation. *Symmetric* filters have linear phase. Except for the Haar basis, all real orthonormal wavelet bases with compact support are asymmetric. Using compactly supported orthonormal wavelets (other than Haar) symmetry and exact reconstruction are incompatible requirements if the same filter is used for reconstruction and decomposition. Biorthogonal wavelet bases uses two dual (symmetric) wavelet filters. Linear filters are wanted in order to avoid phase distortion around edges.

There is a link between the *regularity* of Ψ and the multiplicity of the zero at $\xi = 0$ of $\hat{\Psi}$, where $\hat{\Psi}$ denoted the Fourier transform of Ψ.

Definition. *Lipschitz regularity* [56]. A function f is bounded and uniformly *Lipschitz* α over R if

$$\int_{-\infty}^{\infty} |\hat{f}(\omega)|(1+|\omega|^{\alpha})d\omega < \infty \tag{3.18}$$

These zeros are important because they "kill" aliasing components when cascading the filters. Regularity is concerned with differentiability. In the Fourier domain the existence of derivatives is related to the decay of the Fourier spectra. For fixed support width of ϕ, ψ i.e. for fixed length of the filters in the associated subband coding scheme, the choice with maximum vanishing moments for ψ is different from the maximum regularity choice of h_n . Antonini et al. [3] found that for wavelet coding, the regularity of the reconstruction wavelet $\tilde{\psi}$ is important in a wavelet compression scheme. For the same number of vanishing moments for ψ the scheme with most regular $\tilde{\psi}$ performs the best. Increasing regularity of $\tilde{\psi}$, even at the expense of the number of vanishing moments of ψ performs better. For comparable regularity of $\tilde{\psi}$ the scheme with the largest vanishing moment for ψ is likely to perform best.

The situation is slightly different in a WP coding scheme but it still remains that if ψ has *k* moments which are zero then the coefficients $< \psi_{m,n}, f >$ will represent the features of *f* which are *k* times differentiable with a high compression potential, many coefficients will be close to zero.

The *smoothness* of a wavelet is regarded as an important property. Since images are mostly smooth except for occasional edges it is argued [3] that an exact reconstruction subband coding schema for image analysis should correspond to an orthonormal basis with a reasonably smooth mother wavelet. However, the frequency domain criteria such as constraints on high frequency components, are less important to image compression applications than time domain criteria such as ringing.

The *compact support* of a filter kernel is vital to the time frequency resolution. It is therefore important to have filters that are well localized in space (in time for time signals). The fact that the wavelet is not smooth, such as db4, does not exclude the representation of smooth functions. The right linear combination of these unsmooth wavelets is very good for representing smooth functions as the irregularities cancel out. Compact and unsmooth functions are efficient in representing functions with edges and other discontinuities but smooth functions are not compact.

During the years of the development of the wavelet theory new filters have appeared, fullfilling the demands from the wavelet theory, and have given new life to older signal processing algorithms. Below the filters used in the triplet database are presented.

Haar wavelet is a compactly supported wavelet, the oldest and the simplest wavelet. The scaling function $\phi = 1$ on [0 1] and 0 otherwise. The wavelet function $\psi = 1$ on [0 0.5[, $\psi = -1$ on [0.5 1] and 0 otherwise. The Haar wavelet is orthogonal and symmetric but not continuous. The number of vanishing moments for ψ is 1.

In the group of orthogonal and compactly supported wavelets, the *Daubechies wavelet, symlets* and *coiflets* reside. Their common properties are that ϕ exists and the decomposition is orthogonal, ψ and ϕ are compactly supported and ψ has a given number of vanishing moments. The regularity is poor.

Daubechies wavelets are compactly supported wavelets with extremal phase and a

high number of vanishing moments for a given support width. Associated scaling filter is minimum-phase filter. For order N the support width is 2N-1, filter length is 2N, regularity about 0.2 N for large N. The wavelets are asymmetric and the number of vanishing moments for ψ is N.

Symlets are compactly supported wavelets with least asymmetry and a high number of vanishing moments for a given support width. The associated scaling filter is near linear-phase filter. For order N the support width is 2N-1, filter length is 2N. The filters are nearly symmetric and the number of vanishing moments for ψ is N.

Coiflets are compactly supported nearly symmetric wavelets with high number of vanishing moments for both ϕ and ψ for a given support width. For order N the support width is 6N-1 and filter length is 6N. The number of vanishing moments for ψ is 2N and the number of vanishing moments for ϕ is 2N-1.

The *biorthogonal wavelets* are compactly supported wavelet pairs constructed from B-splines such that symmetry with FIR filters is achieved and desirable properties for decomposition and reconstruction are achieved. ϕ functions exist and the decomposition is biorthogonal, ψ and ϕ both for decomposition and reconstruction are compactly supported, ϕ and ψ for decomposition have vanishing moments. ψ and ϕ for reconstruction have known regularity. For order Nr, Nd (r for reconstruction, d for decomposition) the support width is 2Nr+1 for reconstruction and 2Nd+1 for decomposition. Regularity for the symmetric ψ reconstruction is Nr-1 and the number of vanishing moments for ψ decomposition is Nr.

For the compactly supported biorthogonal spline wavelets symmetry and exact reconstruction are possible with FIR filters while in the orthogonal case this is impossible except for the Haar wavelet.

3.2.5 Measurements of filter effects

We test the performance of the best basis algorithm with a set of the first generation wavelet filters. A listing of filter coefficients can be found in Appendix C.

The filters tested here are Haar, db2, db4, db6, db8, bior1.5, bior2.8, bior3.5, bior6.8, coif1, coif2, coif5, sym2, sym4, sym8, with respective vanishing moments, 1, 2, 4, 6, 8, 5, 8, 5, 8, 2, 4, 10, 2, 4, 8. The results for different combinations are shown in Appendix B and in figures Figure 3.21 and Figure 3.22.

Only the decomposition part is considered here, the theoretically important factor is vanishing moments for the decomposition wavelet. The regularity of the reconstruction wavelet affects the image quality but not the compression ratio. When looking at the images in Figure 3.21, Figure 3.22, and in Appendix B, the vanishing moments do not seem to be the important factor. In fact the performance of the WP coder gets worse with increasing vanishing moments. The theory holds, in practice, only for the wavelet coder with symlet filters.

One explanation is that filtering with the wavelet filter gives large values for image singularities. Most of the filtering in the WP decomposition is done on the detail images. We need a wavelet with more vanishing moments than the numbers of continuous derivatives of the signal for a good compression potential. However the detail bands are not continuously differentiable. Thus we need not have more than one vanishing moment for the wavelet on the detail bands.

The cost functions in Equation 3.12, in Equation 3.13 and in Equation 3.14 give dif-

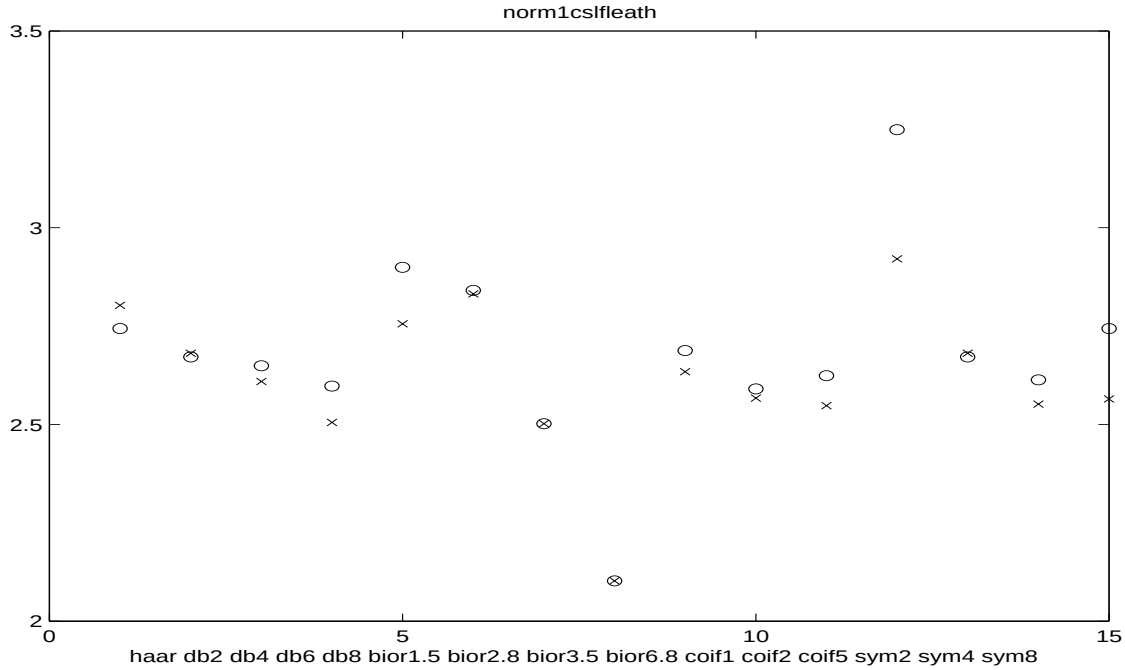


Figure 3.21. The cslfleath image decomposed with varying filters and the cost function n1. The cost of coding the components of the optimal wavelet packet basis is marked by o and the cost of coding the components of the corresponding wavelet basis is marked by x.

ferent performance for the biorthogonal filters. However, the results are worse than the wavelet tree by 0,5 to 5 bpp while the cost function in Equation 3.15, with p=1, follows the wavelet performance more closely. The Haar filter always gives better performance for the WP tree than for the wavelet tree but not necessarily the best performance of the filters. For the wavelet coder the filters db6, bior2.8, coif5 and sym8 perform well with

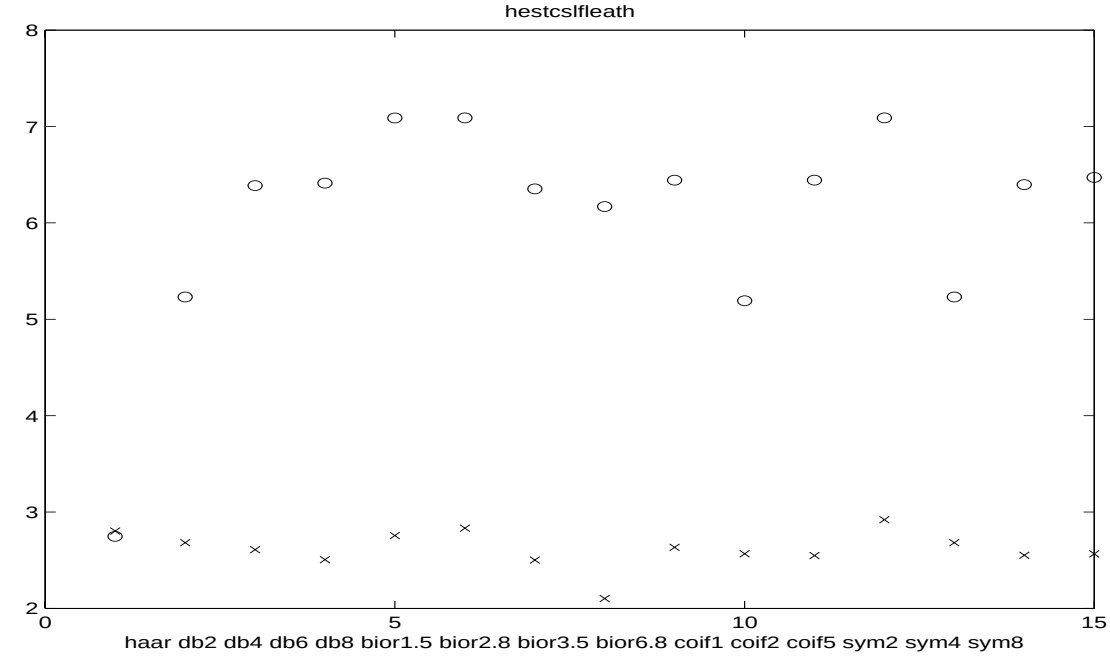


Figure 3.22. The cslfleath image decomposed with varying filters and the cost function hest. The cost of coding the components of the best basis is marked by o and the cost of coding the components of the corresponding wavelet basis is marked by x.

the brickwall image while the bior3.5 is outstanding for the other images.

The symmetry of the decomposing wavelet is more important in a wavelet packet decomposition than in a wavelet decomposition. In the wavelet decomposition only the linear phase of the scaling function matters since the repetition of decomposition is done only on the approximation branch. Daubechies compactly supported wavelets have extremal phase and the highest number of vanishing moments for a given support width. Only the associated scaling filters are minimum-phase filters. This makes them not as suitable for WP decomposition. Symlets are least asymmetric and compactly supported having the highest number of vanishing moments for a given support width with associated scaling filters near linear-phase making them a little bit more suitable. Both Daubechies compactly supported wavelets and Symlet wavelets have poor regularity. Coiflets are orthonormal and nearly symmetric wavelets with vanishing moments for both Ψ and ϕ. They are much more symmetric than the compactly supported wavelets but at the price of wider support.

3.2.6 Conclusion

After analysing the result of combining 17 images, 9 cost functions and 15 different wavelet filters in all possible ways we come to the conclusion that in the context of lossless image compression we are free to choose cost function by a criterion other than the explicit cost of coding the components. The compression does not vary very much with the choice of cost function. Almost none of the tested cost functions for choosing a basis generally finds a basis better than the wavelet tree for all filters.

The L^1-norm and the $L^{1.99}$-norm have been used as the cost function most likely to match the Laplacian and the Gaussian probability distributions. We have used some norms lying in between to see if any of them possibly match some generalized Gaussian distribution. The main part of the subbands has a distribution very like the Laplacian. There is very little difference in the performance with different values of the p parameter in the L^p-norm. The best choice of cost function is the L^1-norm.

The bases chosen with the Haar filter and different cost functions all perform very close to the wavelet basis performance. When the images are analysed with the bior1.5 filter, a difference of up to 3 bit/pixel between different cost functions is produced.

The fact that cascading of filters is done both on the approximation band and the detail bands causes trouble in a wavelet packet decomposition with wavelet filters. The symmetry of the decomposing wavelet is more important in a wavelet packet decomposition than in a wavelet decomposition where only the linear phase of the scaling function matters since the repetition of decomposition is done only on the approximation branch. Most compactly supported wavelets have extremal phase and a high number of vanishing moments for a given support width. Only the associated scaling filters are minimum-phase filters. This makes them not suitable for wavelet packet decomposition due to the cascading effects and, thus accumulated phase errors in the detail bands. Most of the filtering in the wavelet packet decomposition is done on the detail images which are not continuously differentiable. We need a wavelet with more vanishing moments than the number of continuous derivatives of the signal for a good compression potential. This means that we need only one vanishing moment for the wavelet on the detail bands. The Haar wavelet fulfills this demand.

The conclusion is that the choice of filter is important when we want to minimize the

coding cost. The Haar filter gives the wavelet packet basis better performance in combination with almost all of the cost functions tested here. The Haar wavelet is the only real, compactly supported, symmetric, linear phase wavelet that gives perfect reconstruction. Indeed it is the only filter in this set suitable for wavelet packet transform for compression.

3.3 Performance of the CWP coder

The CWP coder, described in section 2.3.4, is a wavelet packet coder that is based on the SPIHT coder, presented in section 2.2.3, for the bit allocation and quantization step. CWP uses a constrained WP tree because the SPIHT coder is designed for the wavelet tree and the reordering algorithm cannot handle all possible WP trees. A suboptimal basis is chosen as close to the best basis as possible.

We use the CWP coder to show the effects of using an optimal basis compared to the use of a suboptimal basis and also compared to the performance of the corresponding wavelet coder. The coder is implemented with the use of the Haar filter and the *n1* cost function for the WP decomposition as this combination has given better coding performance than the corresponding wavelet coder with the same filter for all images tested in section 3.2. In addition to the analysis in section 3.2, we here take the coding of the basis itself into account. The final step of the SPIHT coder is not implemented. This step consists of entropy coding of the bitstream. Thus we will get higher bitrates than published results from using the SPIHT coder on the same images. Since we use the same coding when comparing the different bases, this absence of the last coding step is expected to be of minor importance.

Lenna

The above SPIHT wavelet coder and the CWP coder are both used to code the Lenna image (512x512 pixels). Both coders use the Haar wavelet, decomposing to level 6. The CWP uses *n1* as cost function. Figure 3.23 shows the result. The suboptimal tree of the CWP coder is seen in Figure 3.24. There are virtually no differences between the CWP coder and the SPIHT coder. This is expected as the Lenna image contains mixed textures. Thus the WP algorithm has no chance to find a common resonance in the image.

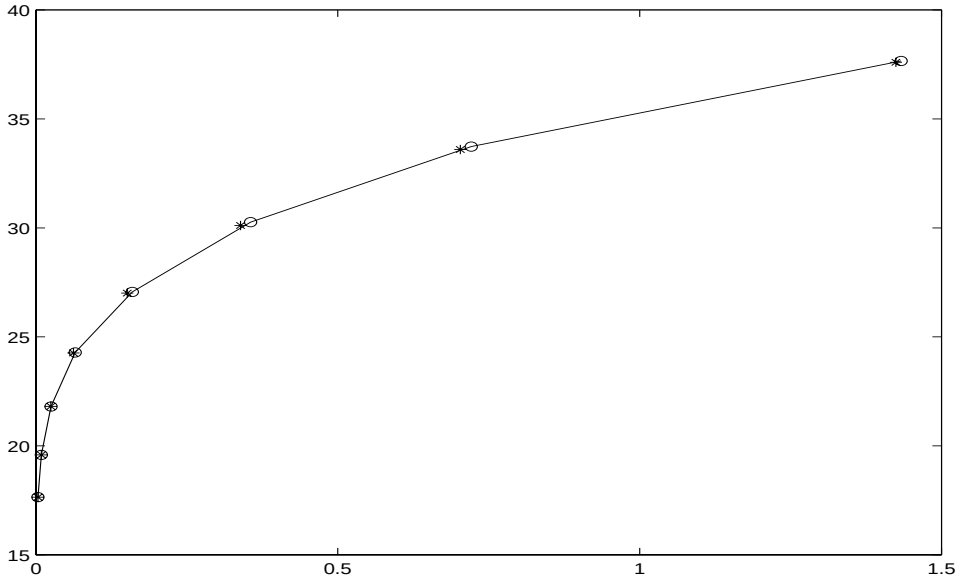


Figure 3.23. PSNR as a function of bpp for the Lenna images, ***** is wavelet coder and **o** is wavelet packet coder.

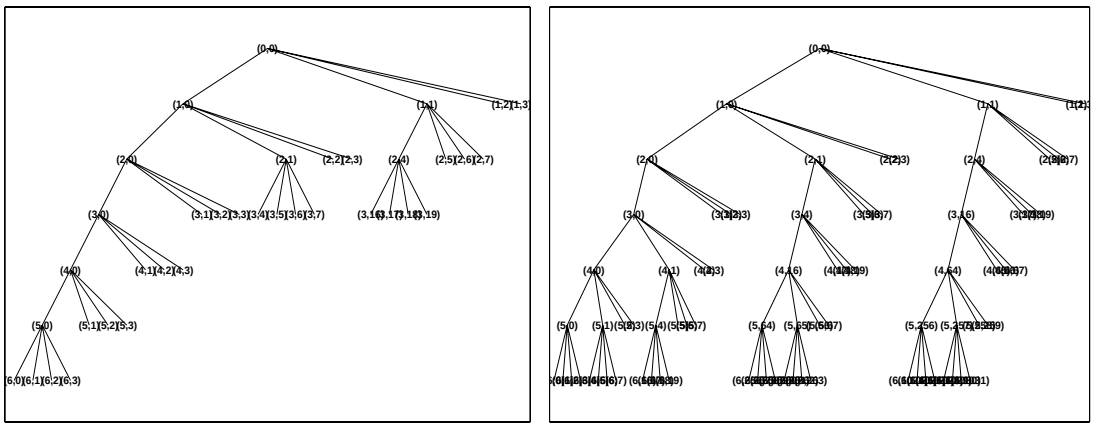


Figure 3.24. Left: the constrained tree used in the SPIHT coder, right: the unconstrained wavelet packet tree representing the best basis chosen for the actual triplet.

Bark

Figure 3.25 shows the corresponding results for the Bark image. Only marginal difference between the CWP coder and the SPIHT coder can be seen. Also for the bark image a constrained, suboptimal tree is used, similar to the wavelet tree in Figure 3.26.

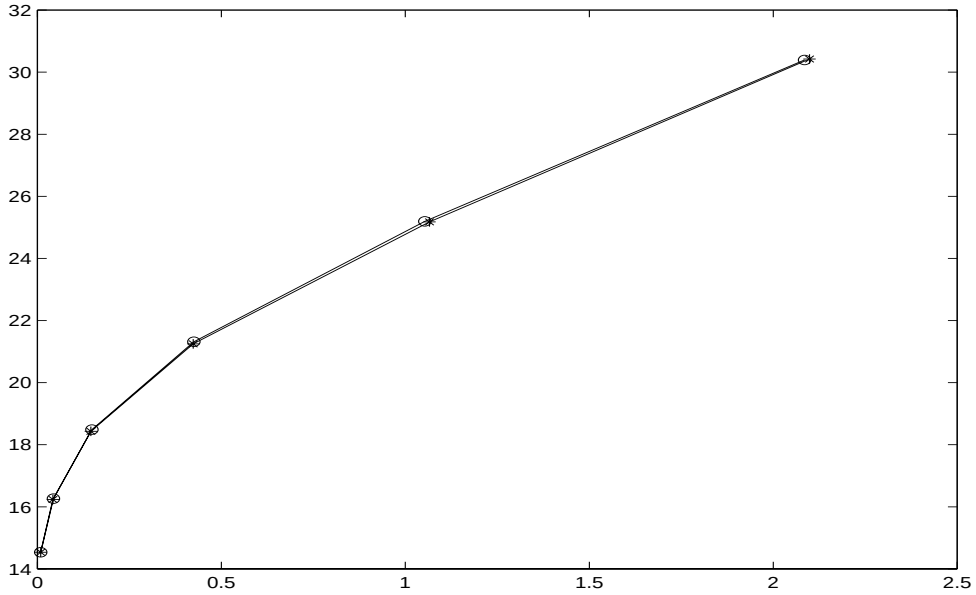


Figure 3.25. PSNR as a function of bpp for the bark images, ***** is wavelet coder and **o** is wavelet packet code

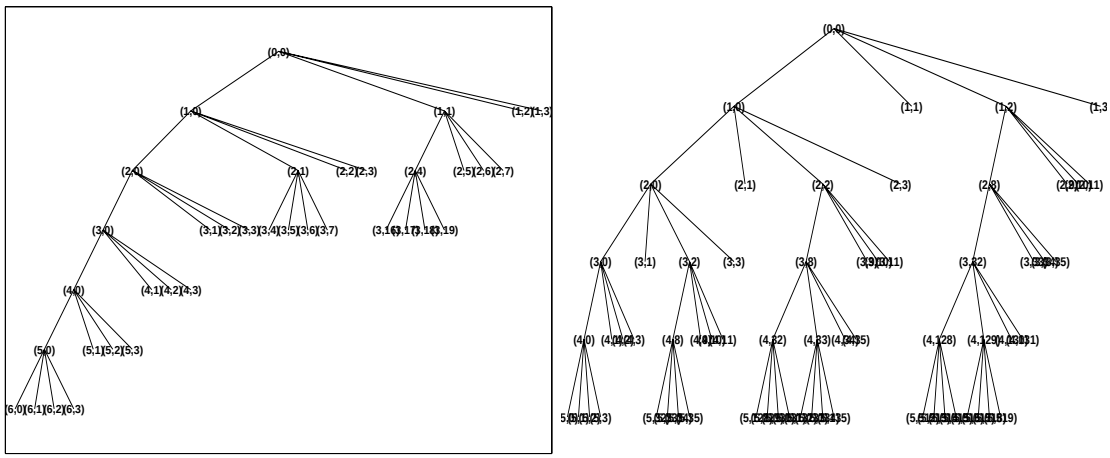


Figure 3.26. Left: the constrained tree used in the SPIHT coder, right: the unconstrained wavelet packet tree representing the best basis chosen for the actual triplet.

Grass

The results in Figure 3.28 show the degradation in PSNR when coding the grass image.

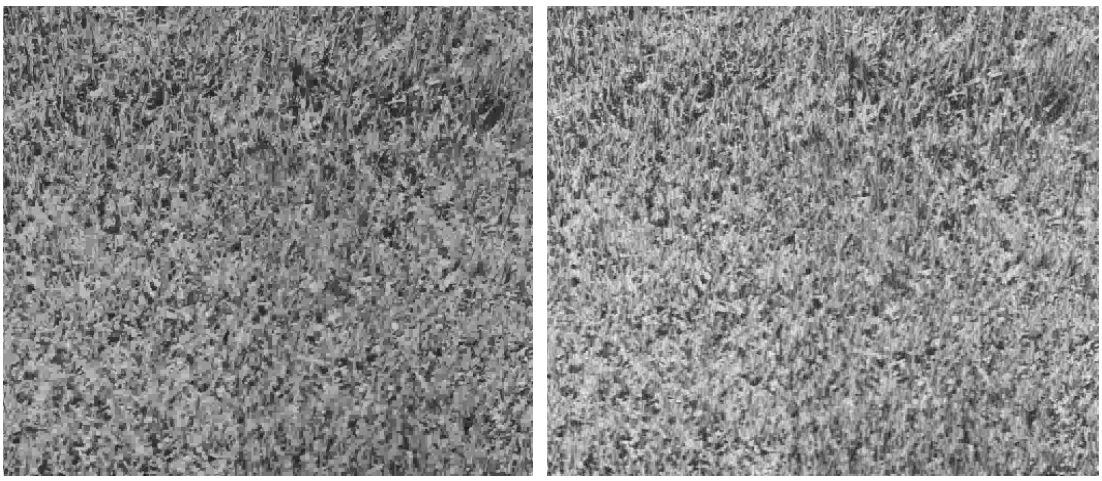


Figure 3.27. To the left: grass coded with SPIHT at 0.7593 bpp, psnr=19.9694. To the right: grass coded with constrained CWP at 0.7499 bpp, psnr=20.1663.

Although the constrained tree for the grass image is not exactly the same as the best basis tree the constrained tree is more similar to the best tree than to the wavelet tree and captures most of the resonance (Figure 3.29). The visual structure is kept at lower bitrates for the WP coder than for the wavelet coder, Figure 3.27.

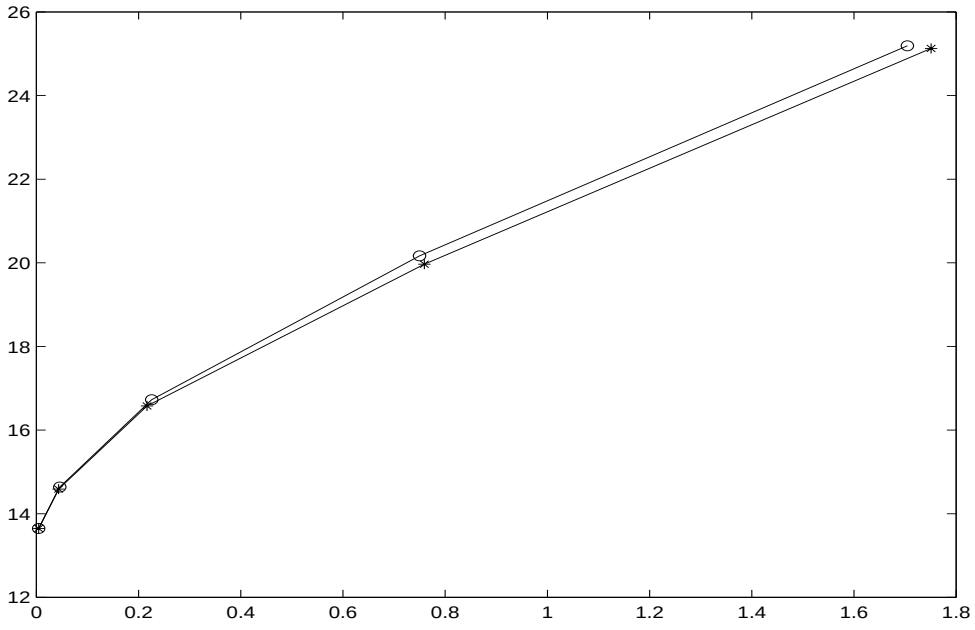


Figure 3.28. PSNR as a function of bpp for the grass images, ***** is wavelet coder and **o** is wavelet packet coder.

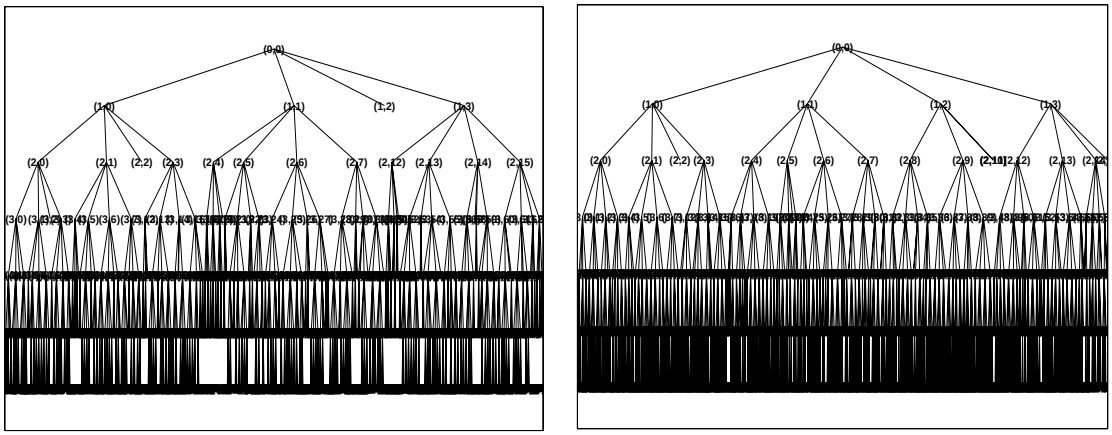


Figure 3.29. Left: the constrained tree used in the CWP coder, right: the unconstrained wavelet packet tree representing the best basis chosen for the actual triplet.

Brickwall

Figure 3.34 shows the corresponding results for the brickwall image.

For the brickwall image the CWP coder uses the best basis (Figure 3.30). A clear difference is seen both visually and in the PSNR, Figure 3.34. The best basis captures the resonance which is better represented by the CWP coder. From the reconstructed images in Figure 3.31 to Figure 3.33 we see the different degradation in visual performance for the SPIHT coder and for the CWP coder at low bitrates. The typical structure for the texture is kept longer for the WP coder.

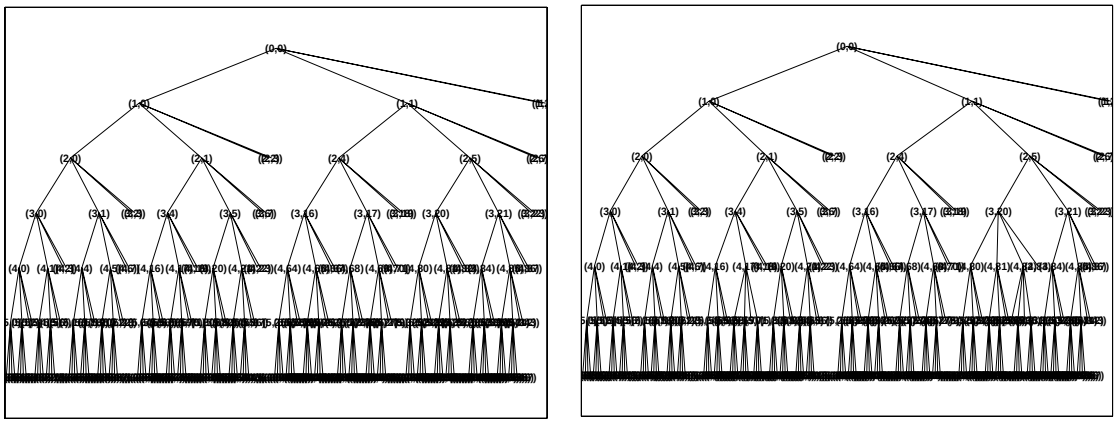


Figure 3.30. left: the constrained tree used in the CWP coder, right: the unconstrained wavelet packet tree representing the best basis chosen for the actual triplet.

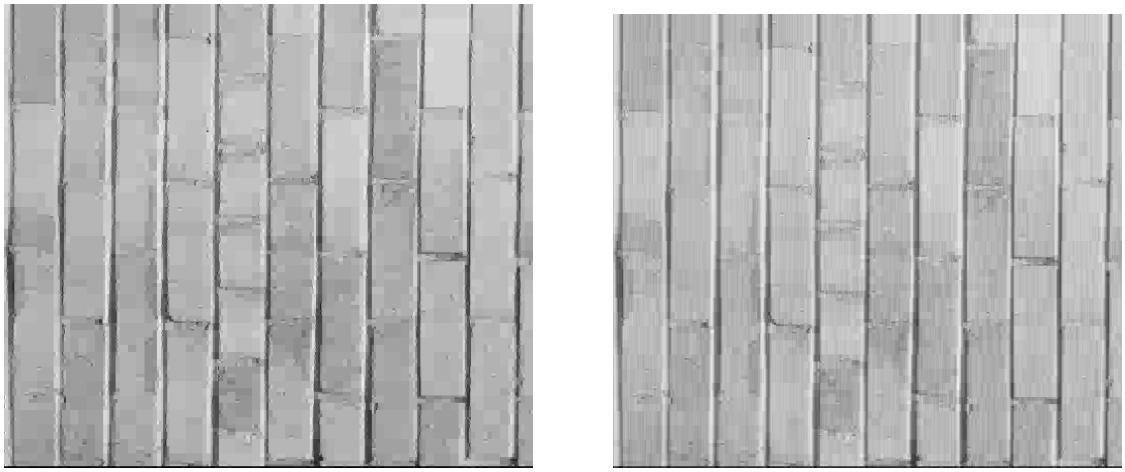


Figure 3.31. To the left: brickwall coded with SPIHT at 0.1621 bpp, PSNR=23.9492. To the right: brickwall coded with constrained CWP at 0.1443 bpp, PSNR=24.3551.

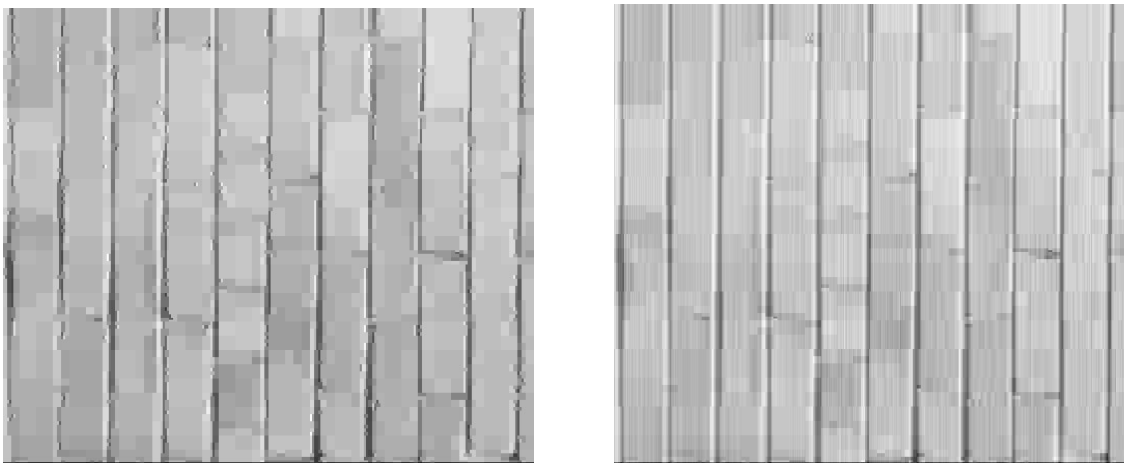


Figure 3.32. To the left: brickwall coded with SPIHT at 0.0631 bpp, PSNR=22.0631. To the right: brickwall coded with constrained CWP at 0.0624 bpp, PSNR=22.7652

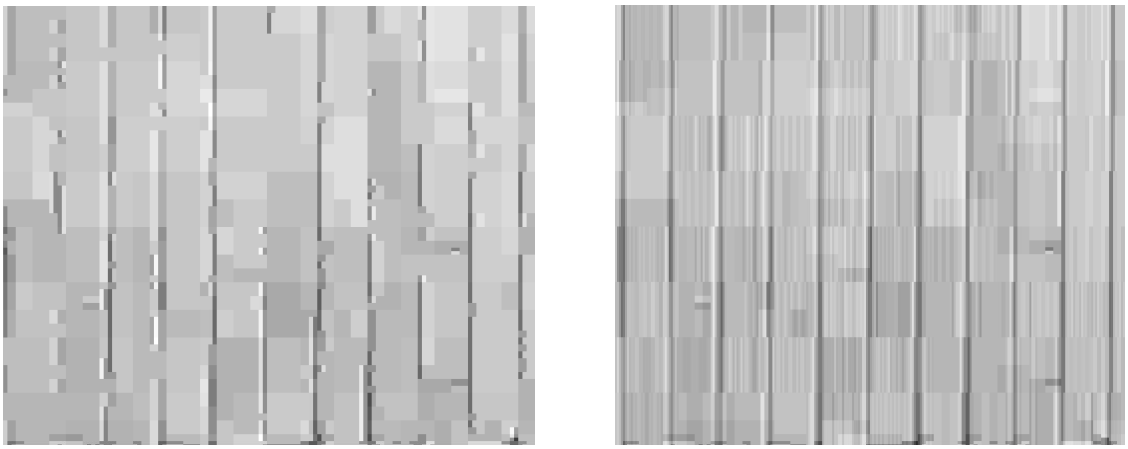


Figure 3.33. To the left: brickwall coded with SPIHT at 0.0215 bpp, PSNR=19.9064. To the right: brickwall coded with constrained CWP at 0.0274 bpp, PSNR=20.9682.

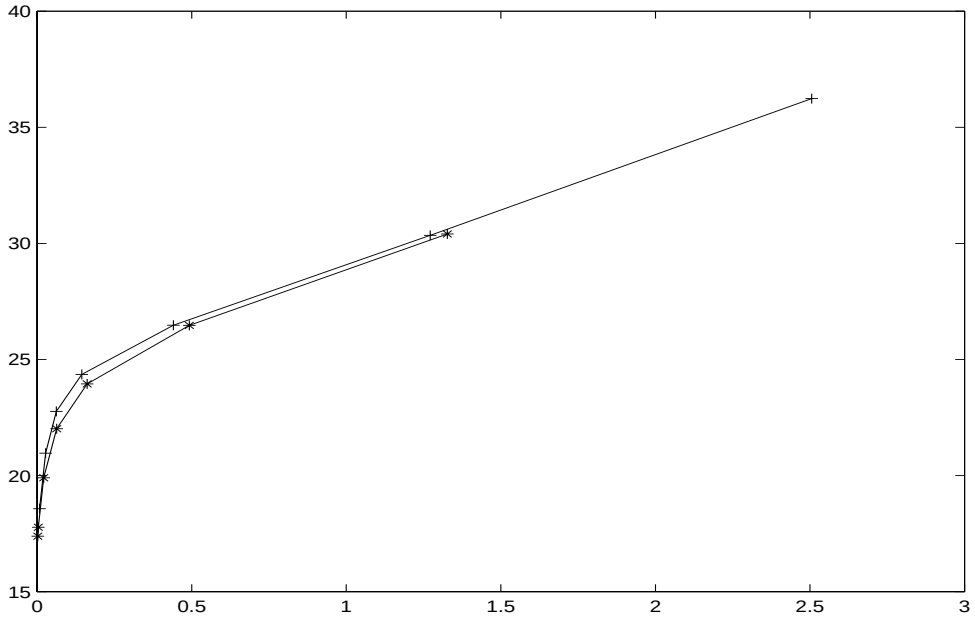


Figure 3.34. PSNR as a function of bpp for the brickwall images, * is wavelet coder and + is wavelet packet coder.

3.3.1 Conclusion

We see that the visual performance as well as PSNR is better when we use an optimal basis rather than a suboptimal basis or a wavelet basis. When the CWP coder is required to use a suboptimal basis the performance is only slightly better than the wavelet coder. All CWP coded images have better quality than the SPIHT coded images. Also, visually the CWP coder gives a better result.

3.3.2 Note on the choice of filter

The Haar filter yields poor reconstruction when used at low bitrates. The PSNR values and the visual performance would definitely be much better with the use of another filter. Why then do we use this filter? The careful reader has already noted some of the reasons from the previous text in this thesis. From section 3.2.4 we note that the Haar filter has the same behaviour for all the images, even if it is not the best filter. From section 3.2.5 we learned that the Haar wavelet is the only real, symmetric, linear phase wavelet for perfect reconstruction out of our set of filters. With this filter we do not need to care about the phase distortion, which can be large for some other filters when cascading on the detail coefficients, as the filters are designed only for cascading on the approximation coefficients. Another reason is that we take advantage of the reconstruction distortion to see the visual differences between the coders.

3.4 Calculated decision rule

In this section we will deduce the decision rule analytically for simple image models, filter and cost function. We do this analysis in one dimension. The extension to two dimensions is straightforward following the separable filtering. These rules are developed for a Gaussian source, Laplacian source, and generalized Gaussian source.

3.4.1 System

We will view the image as a wide-sense stationary random process, X, with the autocorrelation function $r_x[k]$. This function, given by Eq. 3.19), describes the time dependence in the process. It is an ensemble mean taken over all realizations of the process (all images) and is defined by the expected value of the random process X at two different times

$$r_x[k] = E(X_t X_{t+k}) \tag{3.19}$$

The deterministic autocorrelation $p[m]$ of a sequence $h[n]$ is a measure of the time dependencies within the sequence of one image

$$p[m] = \langle h[n], h[n+m] \rangle \tag{3.20}$$

where $\langle\ ,\ \rangle$ denotes the inner product. In an ergodic [74] model this is expected to estimate the autocorrelation function. The estimate of the statistical autocorrelation function for a weak sense stationary sequence can also be computed by the inverse Fourier transform of the periodogram.

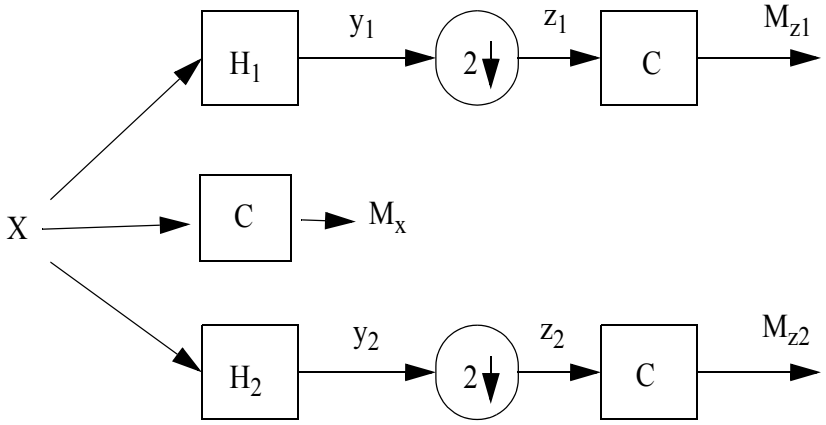


Figure 3.35. Cost measurements before and after splitting.

The situation we want to analyse is described in Figure 3.35. We want to compare the cost $M_{z_1} + M_{z_2}$ of the result from highpass/lowpass filtering and downsampling a signal to the cost M_x of the input signal. The subband is to be split if

$$M_{z_1} + M_{z_2} < M_x \tag{3.21}$$

With the Haar filter we have

$$y_1[n] = \frac{1}{\sqrt{2}}(x[n] + x[n-1]) \tag{3.22}$$

$$y_2[n] = \frac{1}{\sqrt{2}}(x[n] - x[n-1]) \tag{3.23}$$

$$z_1[n] = y_1[2n] = \frac{1}{\sqrt{2}}(x[2n] + x[2n-1]) \tag{3.24}$$

$$z_2[n] = y_2[2n] = \frac{1}{\sqrt{2}}(x[2n] - x[2n-1]) \tag{3.25}$$

By definition

$$r_x[i] = E\{X[k]X[k-i]\} \tag{3.26}$$

we have the autocorrelation functions $r_{z_1}[k]$ and $r_{z_2}[k]$ by

$$r_{z_1}[i] = E\{z_1[k]z_1[k-i]\} = \frac{1}{2}(2r_x[2i] + r_x[2i+1] + r_x[2i-1]) \tag{3.27}$$

and

$$r_{z_2}[i] = E\{z_2[k]z_2[k-i]\} = \frac{1}{2}(2r_x[2i] - r_x[2i+1] - r_x[2i-1]) \tag{3.28}$$

giving the variances as

$$\sigma_{z_1} = r_{z_1}[0] - m_{z_1}^2 = r_x[0] + r_x[1] - 2m_x{}^2 \tag{3.29}$$

$$\sigma_{z_2} = r_{z_2}[0] - m_{z_2} = r_x[0] - r_x[1] \tag{3.30}$$

We choose as our first source model the correlations between neighbours described by the

autocorrelation function

$$r_x[k] = \begin{bmatrix} \sigma_x^2 \rho + m_x^2, |k| = 1 \\ \sigma_x^2 + m_x^2, k = 0 \\ m_x^2, |k| > 1 \end{bmatrix}, \tag{3.31}$$

and secondly we use a Markov source model with autocorrelation function

$$r_x[k] = \begin{bmatrix} \sigma_x^2 \rho^{|k|} + m_x^2, |k| > 0 \\ \sigma_x^2 + m_x^2, k = 0 \end{bmatrix}. \tag{3.32}$$

With these signal models we have the variances for both signal models:

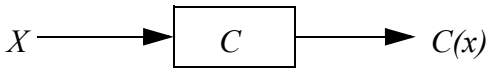
$$\sigma_{z_1}^2 = \sigma_x^2(1 + \rho) \tag{3.33}$$

$$\sigma_{z_2}^2 = \sigma_x^2(1 - \rho) \tag{3.34}$$

This result is reasonable since the Haar filter only considers two neighbouring pixels. The Haar filter makes no difference between the two models.

3.4.2 Applying the cost function

Applying the cost function $C = g(x)$ to the input signal x we get a cost measure $C(x)$ of X



We are interested in the measure of the whole signal or subband, thus the cost function output M_X, assuming that the cost function is additive, is given by

$$M_X = \sum_i C(x_i) \tag{3.35}$$

Apart from a normalization constant, depending on the signal size, this is the mean of the signal under the influence of the cost function. This can be expressed by the expected

value of a random process X representing our signal.

$$E\{g(X)\} = \int g(x)p(x)dx \tag{3.36}$$

where *p(x)* is the probability density function.

One cost function that lends itself easily to analytic computation is the norm $|x|$. Fortunately, it has proven to work well together with the Haar filter as shown in section 3.2. The cost associated with the input signal is given by

$$M_G = E\{|X|\}_G = \int_{-\infty}^{\infty} |x| \frac{1}{\sigma_x\sqrt{2\pi}} e^{-\frac{(x-m_x)^2}{2\sigma_x^2}} dx = \sqrt{\frac{2}{\pi}}\sigma_x + m_x \tag{3.37}$$

for a Gaussian distributed process, and

$$M_L = E\{|X|\}_L = \int_{-\infty}^{\infty} |x| \frac{1}{\sigma_x\sqrt{2\pi}} e^{-\frac{|x-m_x|}{2\sigma_x^2}} dx = \frac{\sigma_x}{\sqrt{2}} e^{-\frac{\sqrt{2}|m_x|}{\sigma_x}} + |m_x| \tag{3.38}$$

for a Laplacian distributed process.

Following [54], we have an expression for the same integral with the generalized Gaussian source (see Equation 3.9).

$$M_{GG} = \int_{-\infty}^{\infty} |x| K e^{-(|x|/\alpha)^\beta} dx = 2K\frac{\alpha^2}{\beta}\Gamma\left(\frac{2}{\beta}\right) \tag{3.39}$$

where α and β are extracted from the signal, for details see [23].

After filtering and subsampling we get the following expressions for the expected costs for a Gaussian source.

$$M_{Gz_1} = \sqrt{\frac{2}{\pi}}\sigma_x\sqrt{(1+\rho)} + \sqrt{2}m_x \tag{3.40}$$

and

$$M_{Gz_2} = \sqrt{\frac{2}{\pi}}\sigma_x\sqrt{(1-\rho)} \tag{3.41}$$

Splitting is done when

$$\frac{N}{2}M_{Gz_1} + \frac{N}{2}M_{Gz_2} < NM_G \tag{3.42}$$

which leads us to the following decision rule for both the source models and a Gaussian distribution

$$\sqrt{\frac{2}{\pi}}\frac{\sigma_x}{2}(\sqrt{1+\rho} + \sqrt{1-\rho} - 2) + (\sqrt{2}-1)m_x < 0 \tag{3.43}$$

For mean values $m_x < -5.3$, for ρ ranging from 0 to 1 and the variance σ_x^2 ranging from 0 to 100 the decision is to always split.

For the Laplacian distribution the decision rule for both source models is described by

$$M_{Lz_1} = \frac{\sigma_x\sqrt{(1+\rho)}}{\sqrt{2}}e^{-\frac{2|m_x|}{\sigma_x\sqrt{(1+\rho)}}} + \left|\sqrt{2}m_x\right| \tag{3.44}$$

and

$$M_{Lz_2} = \frac{\sigma_x\sqrt{(1-\rho)}}{\sqrt{2}}e^{-\frac{2|m_x|}{\sigma_x\sqrt{(1-\rho)}}} \tag{3.45}$$

The split decision rule is written

$$\frac{N}{2}M_{Lz_1} + \frac{N}{2}M_{Lz_2} < NM_L \tag{3.46}$$

leading to

$$\frac{\sigma_x}{2\sqrt{2}}\left(\sqrt{1+\rho}\,e^{-\frac{2|m_x|}{\sigma_x\sqrt{(1+\rho)}}}+\sqrt{1-\rho}\,e^{-\frac{2|m_x|}{\sigma_x\sqrt{(1-\rho)}}}-2e^{-\frac{\sqrt{2}|m_x|}{\sigma_x}}\right)+(\sqrt{2}-1)|m_x|<0 \quad (3.47)$$

In Figure 3.36 we present plots of Equation 3.47 for ρ ranging from 0 to 1 and the variance σ_x^2 ranging from 0 to 100, for different mean values m_x. The black area describes for which values of ρ and σ the band will be split. A mean value of 0 indicates that a split is made.

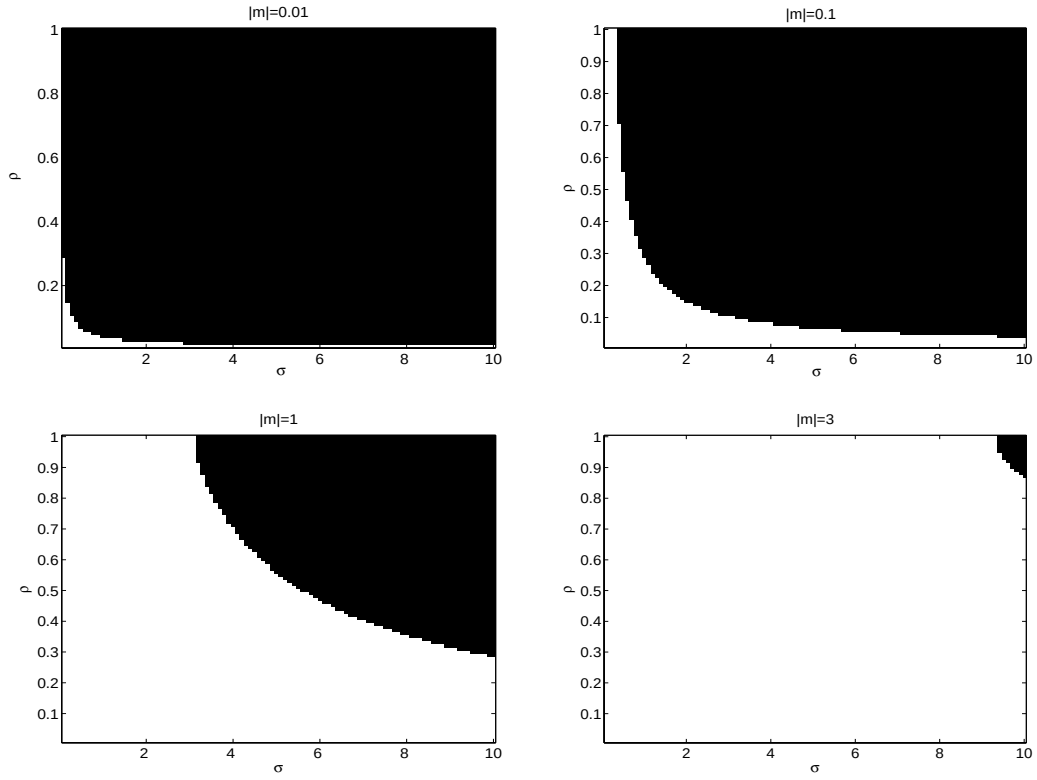


Figure 3.36. The decision rule plotted for a Laplacian process with different mean values, for ρ between 0 and 1 and σ between 0 and 100. The black area describes for which values of ρ and σ the band will be split.

We have produced decision rules for the split/no-split decision to be applied to every subband. These rules uses the autocorrelation function of the mother band to calculate the effect of splitting the band without actually performing the split. This can be used both top down in a greedy search and bottom up like the best basis search. For a typical image where the mean is positive, the first level will be split. The approximation band will always have a positive mean value and is thereforee more likely to split than the detail bands. This is probably the main reason why the wavelet decomposition comes so close to the optimal solution.

The brickwall image is one of the images with an autocorrelation function that does not let itself be modelled by the image models used here. Grass, on the other hand, has an autocorrelation function that better fits the models used in the calculations. With the assumption that the approximation band always will be split, we will use the calculated decision rules in a top down fashion on the grass image. The autocorrelation function for

each subband, in both horizontal and vertical direction, is plotted together with the corresponding values of mean, variance and correlation coefficient in Figure 3.37. These values are fed into the decision rules to achieve the second-level split. The calculated decision rule for an image with a Gaussian pdf suggests to split every subband while the decision rule for an image with a Laplacian pdf suggests to keep every subband. The truth is probably somewhere in between and the decision rule for an image with a generalized Gaussian pdf could be used.

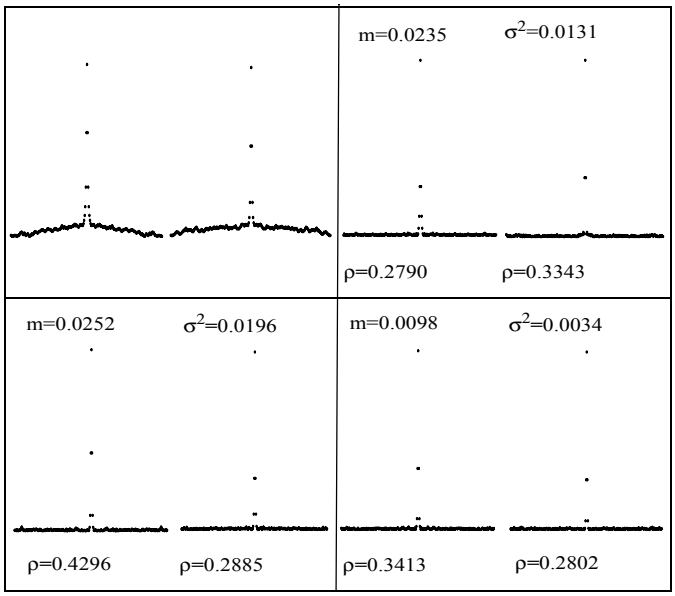


Figure 3.37. The first level decomposition of the Grass texture, with plots of the ACF and the values.

3.4.3 Conclusion

The calculated decision rule does not work with an image with an autocorrelation function that cannot be modelled by the image models used in the calculations, but could find a basis for an image with a reasonable autocorrelation function. The Gaussian and Laplacian pdf models used here are to the two extremes when modelling the amplitude distribution of the image. Using the deneralized Gaussian can lead to a useful method for taking the decision of split in a top-down manner, but this is left to future work.

3.5 Summary

This chapter deals with the problem of finding the basis that is best for representing the image in a compression application. The focus is on the transform in contradiction to the methods presented in Chapter 2. We start with an unbiased investigation of what happens with the compression ratio when different images are wavelet packet analysed using combinations of different filters and cost function. The compression ratio is estimated by the entropy of the coefficient matrix. This is motivated by the assumption of a memoryless source generating the image. This assumption is often used in the literature when try-

ing to adapt methods to the image statistics only by taking the probability density function into account. The results are discussed with respect to the properties of the filters and cost functions used. One filter and one cost function are chosen and put into an implementation of a wavelet packet coder presented in Chapter 2. This is in order to view the effect of using a basis which is optimal with respect to image, filter and cost function.

Next we analytically calculate the decision rule based on a selection of filter and cost function. The image is modelled by the spatial dependencies described by the autocorrelation function. The probability density function is set to be Gaussian or Laplacian but could be adapted to the image by the use of the generalized Gaussian.

We find that the choice of filter is more important to the compression performance than the choice of cost function. Due to the cascaded filtering process applied on the detail bands, the phase effects disqualify most wavelet filters for the wavelet packet use. The needed number of vanishing moments of the filters depends on the differentiability of the image. For the detail bands we do not need more than one vanishing moment. The Haar filter is in fact the only filter out of our filter set that can be used, although it has poor reconstruction performance. We also note that a wavelet packet basis is better than the wavelet basis only when the optimal basis is used. Any sub-optimal basis, even close to the optimal, gives performance close to that of a wavelet basis using the same filter.

Chapter 4

Discussion - Wavelet packet coding

The wavelet packet part of this thesis starts with a survey of different wavelet and wavelet packet image coding methods that evolved over the years. Adelson and others provide a start through pyramid image coding [1,2,12]. Even though the pyramid was created by a low-pass filtering process by successive convolutions of the images with Gaussian-like kernels, the difference between the resulting images was the equivalent to convolving the image with an appropriately scaled Laplacian weighting function [1,2]. In [12] the theory is expanded to including band-splitting transforms using quadrature mirror filters (QMF) and Haar basis. The QMF approach was also treated by Woods & O'Neil [94], in which the expansion to the multi-dimensional subband coding of [89] to image subband coding appeared. With Mallat [54,55] the connection between the Laplacian pyramid image coding, subband coding and wavelet theory [20,21] was established.

Many ideas evolved for signal analysis and image processing applications such as image coding. The wavelet packet transform [16,17] which was a natural extension of the wavelet analysis was applied to images in [92]. Herley et al.'s work [32,33] on tilings of the time frequency plane to adapt the wavelet packet basis to time varying structures was applied to images in [34], and further developed to joint dual double trees in [82]. The latter method can tile the time frequency plane in ways that the wavelet packet method cannot and thus can reach a more efficient decomposition of the signal. Ramchandran et al. extended this [71]. While these tiling are still rectangular, another approach was taken by Baraniuk & Jones [5,6] with the introduction of the chirplet transform which gives non-rectangular tilings of the time frequency plane.

Recently the multiwavelet and multiwavelet packets were introduced in image compression applications [57]. These methods use more than one scaling function and wavelet function to reach better results.

Apparently there are many variations on the theme of finding the best representation. In this part we concentrated on the wavelet packet transform and dissect the problems of the method that affects the result and which cannot be solved by the extensions described

above.

The wavelet compression methods have developed into use in standards while the wavelet packet algorithms still struggle with problems. Understanding the Triplet Problem, treated in Chapter 3, is essential for the development of the wavelet packet methods. Almost everyone using wavelet packet methods refers to the fact that the wavelet packet transform is better than the wavelet transform on images with high spatial frequency patterns, such as the Barbara image (in Figure 5.38). This is true to some degree. It was early noted [15] that the wavelet packet transform can only represent the resonance in the image. A high spatial frequency pattern is a resonance and the Barbara image is full of that, with different orientation. Each orientation of this pattern generates a different wavelet packet basis when the algorithm is applied on the pure pattern. The Barbara image is, as most natural images, a mixture of different spatial frequency patterns, or textures, and thus, the wavelet packet transform fails to find the very best representation for the image by one basis alone. However, this particular image contains textures similar enough to make the wavelet packet transform outperform the wavelet transform and the image is a popular choice for the testing of wavelet packet image compression algorithms.

The problem how to decompose different parts of an image differently was treated by [34] and [82] as discussed above. In the wavelet part of this thesis the work ignore this problem and is concentrated on pure texture images. The problem of proper choice of filter and cost function still remains and this is treated in this thesis. The reviewed algorithms presented in Chapter 2 are typical in the sense that they focus on the quantization, bit allocation, and entropy coding steps of the coding chain shown in Figure 1.10. The problem of choosing the basis is often not treated. In the wavelet case one can argue that the basis is fixed but the coding result change with the choice of filters. Most wavelet packet image coding algorithms use wavelet filters without any motivation other than that it is a good filter for wavelet image coding. In this thesis we have shown experimentally the effects of using wavelet filters in both wavelet and wavelet packet coding of texture images. The wavelet filters are designed for iterative use only on the low-pass part of the signal. Only the low-pass filter corresponding to the scaling function has properties appropriate for cascading the filters. When using the high-pass filter corresponding to the wavelet in a cascading manner, errors are easily introduced in the image. The filters treated in this thesis are the first generation wavelet filters. Using these are sufficient to show the differences between the wavelet transform and the wavelet packet transform regarding the needs for proper filters. We find that the choice of filter is more important to the compression performance than the choice of cost function. Due to the cascaded filtering process applied on the detail bands, the phase effects disqualify most of the filters for wavelet packet use. For the detail bands we do not need more than one vanishing moment. The Haar filter is in fact the only filter out of our test set that can be used, although it has poor reconstruction performance.

The focus of the first part of this thesis has been to understand the behaviour of the wavelet packet transform rather than finding a new good filter set. However, an embryo of basic criteria for such a filter set is outlined in that both low-pass and high-pass filter should be linear phase filters and the decomposing filter set can be Haar-like while the reconstruction filter set needs to be smoother.

For the wavelet packet case the bases become totally different depending on the image, the choice of filter, and the choice of cost function. When using a proper filter set, like Haar, we found that the choice of cost function is of minor importance for the compression result. The choice of cost function in a wavelet packet application is treated in

[84,85] for speech compression where they examine additive and non-additive cost functions in both top-down and bottom-up search approaches and present a near-best basis selection algorithm. The work is very interesting due to its extensive examination of cost functions in different basis search algorithms and their conclusion that the choice of cost function has significant impact on the result in both time-frequency analysis and compression. However, they used filters which were optimal for the wavelet transform, and thus the result of their work is not valid for our evaluation of the wavelet packet transform.

The best basis method of [17] is a bottom-up search algorithm often stated to find the global optimal basis. This method decomposes the image down to some level and applies the cost function on the nodes from the bottom up to decide whether or not to cut the tree at this node. However, we have found in this thesis that the best basis method finds different bases when the image is decomposed to different levels.

In this thesis a method of analytic calculation of the expected basis has been presented. The method uses the analytic expression of the cost function, the filter parameters and the image statistics by means of two different models for the pixel interdependencies described by the autocorrelation function. This is an approach that differs from using the image or subband histogram only as in [11] for example. The memory free assumption is used in most wavelet coding algorithm, including the best basis algorithm, when the histogram is used for approximation of the probability function. When the image is modelled by its spatial dependencies described by the autocorrelation function we are taking the memory of the source into account. This is to take the wavelet packet coding algorithms one step further towards a useful algorithm.

The examination of wavelet packet methods in this thesis demonstrates the difficulties to utilize these methods for improved image coding. The potential seems to be there, however, it has to be admitted that the problem is not yet solved.

PART TWO

EMPIRICAL MODE DECOMPOSITION

Chapter 5

Empirical Mode Decomposition

The wavelet packet algorithms treated in previous chapters search for the resonance in the signal. One of the purposes for the thorough examination of the algorithms was to find out if the wavelet packet decomposition could serve as the totally adaptive decomposition for any signal. The limitations due to filters and cost functions in combination with different signals made this difficult. The totally different approach of the Empirical Mode Decomposition (EMD) was revealed in [36]. EMD is an adaptive decomposition with which any complicated signal can be decomposed into its Intrinsic Mode Functions (IMF). EMD is an analysis method that in many aspects gives a better understanding of the physics behind the signals. Because of its ability to describe short time changes in frequencies that can not be resolved by Fourier spectral analysis it can be used for nonlinear and non-stationary time series analysis. For the IMF we can have a well-defined instantaneous frequency. The original purpose for the EMD was to find a decomposition which made it possible to use the instantaneous frequency for time-frequency analysis of non-stationary signals. In this method we found the totally adaptive signal decomposition we had been looking for and we expanded the concept of EMD to two dimensions for image analysis and compression purposes [51].

This chapter is devoted to understanding EMD in both one and two dimensions. We explain the algorithms and put the focus on different limitations initialized by different implementation methods.

This chapter begins with a review of the EMD applied to time signals. Previous work often mentions the lack of a mathematical formalism to describe the EMD. The concept is truly empirical. In this work we keep the empirical approach as we expand the method to two-dimensional signals, such as images. Some implementations of the EMD in two dimensions generate a residue with many extrema points. We will give an example of such an implementation and explain its drawbacks. In this chapter we also present an improved method that can decompose the image into a number of IMFs and a residue with none, or with only a few extrema points. This method makes it possible to use the EMD for image

processing. We introduce the concept of *empiquency*, short for empirical mode frequency, to describe the signal oscillations as the traditional frequency concept is not applicable in this context. We also discuss the selection of *significant extrema points* as a tool for noise reduction.

5.1 EMD for time signals

The EMD is an adaptive decomposition with which any complicated signal can be decomposed into a redundant set of signals denoted IMF and a residue. Adding all the IMFs together with the residue reconstructs the original signal without information loss or distortion.

An IMF is characterized by some specific properties. One is that the number of zero crossings and the number of extrema points are equal or differs only by one. Another property of the IMF is that the envelopes defined by the local maxima and minima, respectively, are locally symmetric around the envelope mean. The original purpose for the EMD was to find a decomposition which made it possible to use the instantaneous frequency, defined as the derivative of the phase of an analytic signal, in the time frequency analysis of the signal [36]. Traditional Fourier analysis does not allow a signal's spectrum to change over time. The EMD, however, provides a decomposition method that analyses the signal locally and separates the component holding locally the highest empiquency (see section 5.1.4), from the rest into a separate IMF. Within this IMF both high and low empiquencies can coexist at different times. The key word is *locally*.

In this thesis the Hilbert-Huang Transform is presented in an example for the introduction of the EMD of time signals, and as the origin of EMD. We then leave the HHT and develop the EMD concept further in the time domain and image domain for compression purpose.

5.1.1 Time-frequency analysis with Hilbert-Huang Transform (HHT)

Many methods exist which analyse signals simultaneously in the time- and frequency-domain, wavelets [56] and short-time Fourier transform [14], among others. These methods are based on the expansion of the signal into a set of basis functions where the basis functions are defined by the method. The concept of the EMD is to expand the signal into a set of functions defined by the signal itself, the IMFs. Signal adaptive decomposition by means of Principal Component Analysis (PCA) [29] also expands the signal into a basis defined by the signal itself. The PCA differs from EMD in that it is based on the signal statistics, while EMD is deterministic and is based on local properties. The EMD process allows time-frequency analysis of transient signals for which Fourier based methods have been unsuccessful. Whenever we use the Fourier transform to represent frequencies we are limited by the uncertainty principle. For infinite signal length we can get exact information about the frequencies in the signal, but when we restrict ourselves to analyse a signal of finite length there is a bound on the precision of the frequencies that we can detect. The instantaneous frequency represents the frequency of the signal at one time, without any information about the signal at other times. There exist different definitions for

instantaneous frequency; a common one is to use the derivative of the phase of the signal as described in [36]. Another problem with using the instantaneous frequency is that it only provides one value at each time. A signal usually consists of many intrinsic frequencies and this is where the EMD is used, to decompose the signal into its IMF, where only one frequency component is present at each time so that a well-defined instantaneous frequency can be conputed.

A relatively new technique has been developed to perform general analysis of highly transient time-domain signals, called the Hilbert-Huang Transform (HHT) [36]. It has shown great utility in time-frequency analysis of dispersive, nonlinear, or non-stationary signals and systems. The transform uses the EMD, with which the signal is decomposed into a redundant set of IMFs and a residue. By applying the Hilbert transform to each IMF we get a set of analytical signals representing the input signal. The HHT calculates the instantaneous frequency of each transformed IMF and presents the result as a time-frequency analysis in a Hilbert spectrum plot. Time-frequency analysis by means of HHT, or EMD separately, has been done in several areas, such as analysis of ocean waves, seismology, and one-dimensional analysis of SAR images [36,88,37]. In a study of laser vibrometry data for target classification, the HHT outperforms wavelet analysis and the power spectral density method [61].

Example 5.1. We create a composite chirp by adding together a 2-second long linearly increasing chirp ranging from DC to 300 Hz with a "convex" quadratic chirp of duration 1 second which starts at 25 Hz and increases to 100 Hz, and a "concave" quadratic chirp of duration 1 second which starts at 100 Hz and decreases to 25Hz. The resulting signal is sampled at 1 kHz and is shown in Figure 5.1

Figure 5.2 shows the time frequency plot of the composite chirp signal by the windowed Fourier transform to the left and the corresponding plot of a wavelet packet best basis decomposition of the same signal to the right.

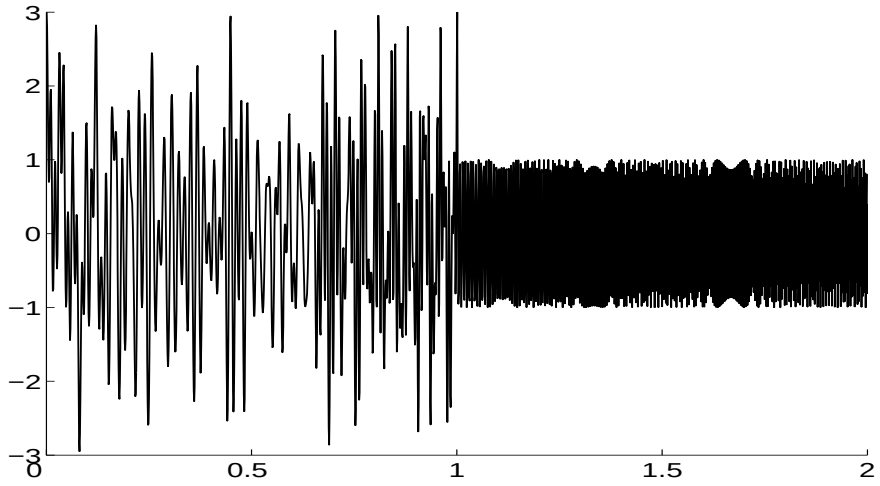


Figure 5.1. Time plot of the composite chirp test signal.

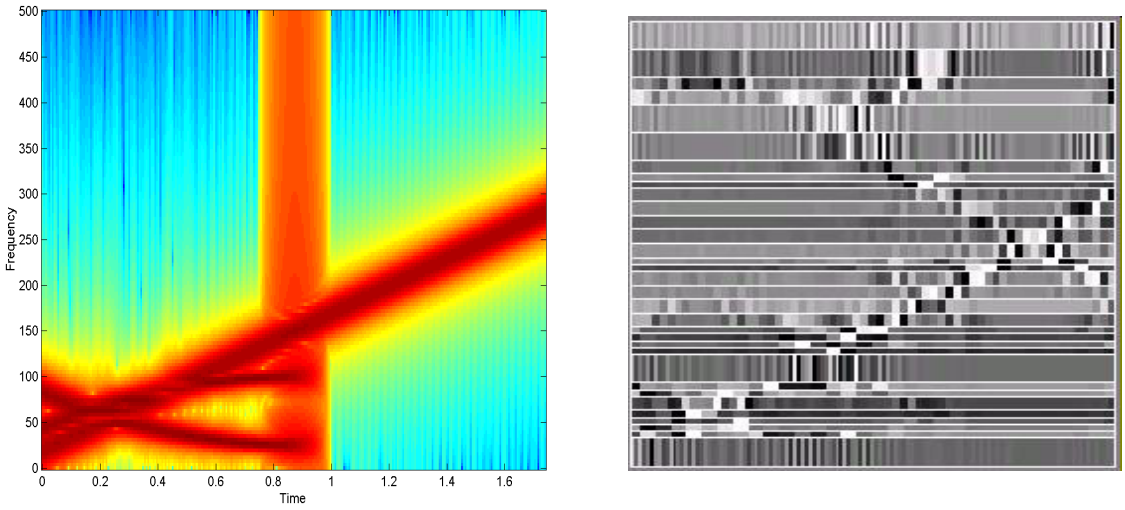


Figure 5.2. Time-frequency plot of the composite chirp signal by the windowed Fourier transform to the left and the corresponding plot of the wavelet packet best basis decomposition of the same signal to the right.

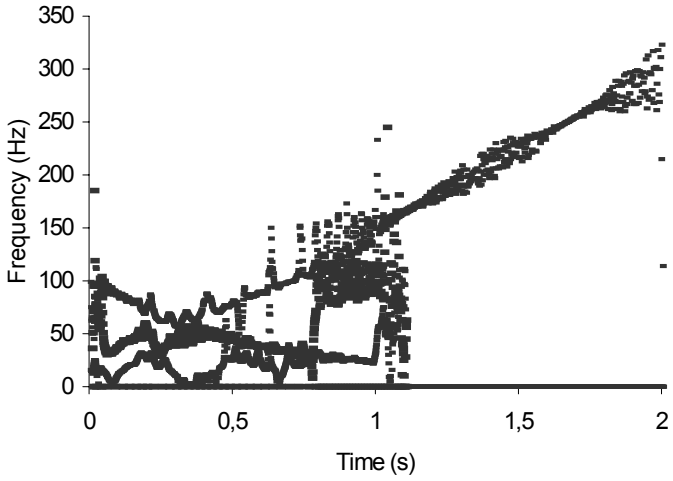


Figure 5.3. The Hilbert spectrum of the composite chirp signal.

The Hilbert spectrum of the test signal is shown in Figure 5.3. It is composed by Hilbert spectra of each IMF of the composite chirp signal shown in Figure 5.15.

Both the *short-time Fourier transform* and the *wavelet packet best basis* representations suffer from the uncertainty principle. In the left plot in Figure 5.2 the signal response is smeared along the frequency axis as a result of the narrow time window and hence a broad frequency window used to analyse the signal. On the right side of Figure 5.2 the wavelet packet best basis representation is plotted. Here all the small boxes are of the same area and the grayscale indicates the correlation between the signal and the analysis filter associated with the box: white indicates high correlation and black indicates low correlation. The form and position of the box is dictated by the *best basis* search algorithm originally described in [17].

The Hilbert spectrum is plotted against the time axis in Figure 5.3, and it shows not only the inter frequencies generated by the construction of the signal, but also the intra frequencies which come from the change of curve form by the frequency change and the addition of signals. This is an advantage with the EMD compared to Fourier methods as it offers more features for the analysis of the signal. Artifacts are also introduced by the use of sampled signals.

5.1.2 Sifting process for finding the IMF

The sifting process to find the IMFs is introduced in [36]. We describe and picture it by a very simple test signal denoted by $x(t)$. It is constructed by the addition of two sine curves and a slowly varying trend, and is visualized by the green curve in Figure 5.4 (a).

To find the first IMF, start with the image itself as input signal $in_{11}(t) = x(t)$. The first index is the IMF number, l=1..L, and the second index is the iteration number, k=1..K, in the sifting process. To find the next IMF, use the residue corresponding to the previously found IMF as input signal $in_{21}(t) = r_1(t)$.

The sifting process to find the IMFs of a signal *x(t)*, comprises the following steps:

(1) Find the positions and amplitudes of all local maxima, and find the positions and amplitudes of all local minima in the input signal $in_{lk}(t)$. These are marked by red and blue dots respectively in Figure 5.4(a)

(2) Create the upper envelope by spline interpolation of the local maxima and the lower envelope by spline interpolation of the local minima, denoted $e_{max}(t)$ and $e_{min}(t)$. These are shown as the red and blue curves in Figure 5.4(b). Cubic spline is used in this example.

(3) For each time instant t, calculate the mean of the upper envelope and the lower envelope.

$$em_{lk}(t) = \frac{e_{max}(t) + e_{min}(t)}{2} \tag{5.1}$$

This signal $em_{lk}(t)$ is referred to as the envelope mean and is shown as the black line in Figure 5.4(b), (c), and (d).

(4) Subtract the envelope mean signal from the input signal (black in Figure 5.4(b) from green in Figure 5.4(b)), yielding the results illustrated by green in Figure 5.4(c).

$$h_{lk}(t) = in_{lk}(t) - em_{lk}(t) \tag{5.2}$$

This is one iteration of the sifting process. The next step is to check if the signal $h_{lk}(t)$ from step (4) is an IMF or not. In the original work [36] the sifting process stops when the difference between two consecutive siftings is smaller than a selected threshold ε. In this thesis the process stops when the envelope mean signal is close enough to zero, as suggested in [51].

$$\left|em_{lk}(t)\right| < \varepsilon \qquad \forall\ t \tag{5.3}$$

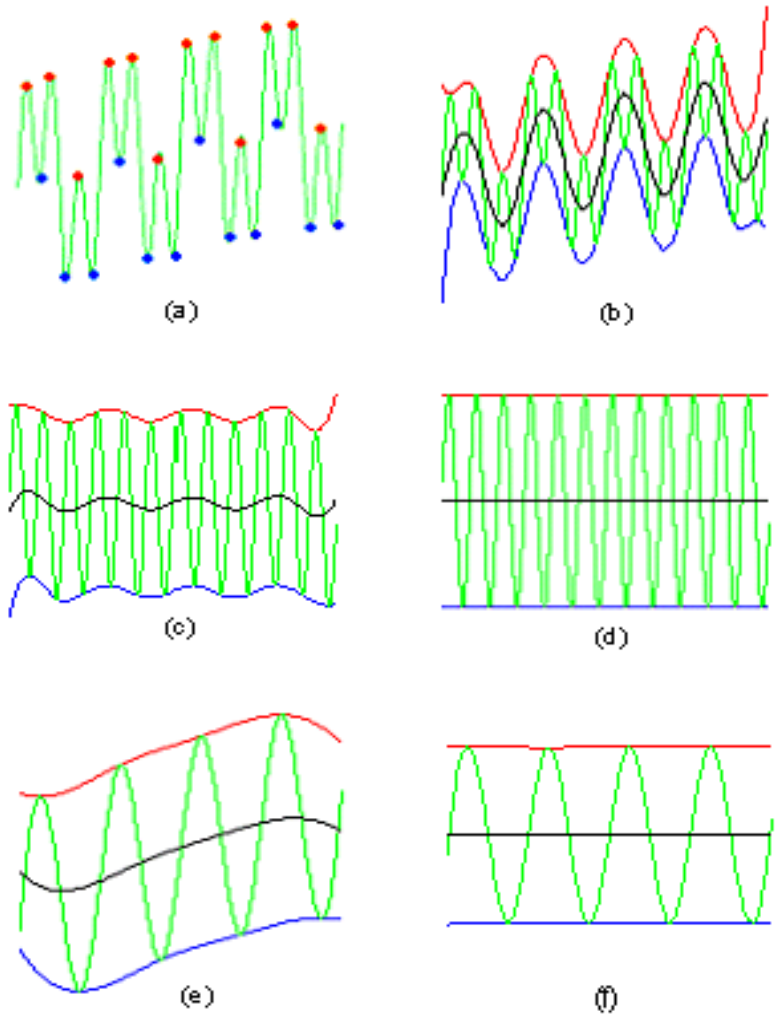


Figure 5.4. a: The test signal with the local maxima marked by red dots and the local minima marked by blue dots. b: The test signal with upper envelope in red, lower envelope in blue and the envelope mean in black. c: The result of one iteration in the IMF sifting process with the upper and lower envelope in red and blue and their mean in black. d: The resulting first IMF with the upper and lower envelope and their mean. e and f: Iteration for the second IMF.

The reason for this choice is that forcing the envelope mean to zero will guarantee the symmetry of the envelope and the correct relation between the number of zero crossings and number of extremes that define the IMF. A slightly more complicated version of this stop criteria is presented in [73], along with a discussion of typical threshold values.

(5) Check if the mean signal is close enough to zero, based upon the stop criterion. If not, iterate by repeating the process from step (1) with the resulting signal from step 4 as the input signal a sufficient number of times.

$$in_{l(k+1)}(t) = h_{lk}(t) \tag{5.4}$$

When the stop criterion is met, $k=K$, the IMF is defined as the last result from (4).

$$c_l(t) = h_{lK}(t) \tag{5.5}$$

After the IMF $c_l(t)$ is found, (here illustrated by the green curve in Figure 5.4(d)), define the residue $r_l(t)$ as the result of subtracting this IMF from the input signal.

$$r_l(t) = in_{l1}(t) - c_l(t) \tag{5.6}$$

The residue is illustrated by the green curve in Figure 5.4(e).

(6) The next IMF is found by starting over from step 1, now with the residue as the input signal.

$$in_{(l+1)1}(t) = r_l(t) \tag{5.7}$$

The second IMF is illustrated in Figure 5.4(f).

Steps (1) to (6) can be repeated for all the subsequent r_j. The EMD is completed when the residue, ideally, does not contain any extrema points. This means that it is either a constant or a monotonic function. The signal can be expressed as the sum of IMFs and the last residue

$$x(t) = r_L(t) + \sum_{j=1}^{L} c_j(t) \tag{5.8}$$

The first IMF captures the highest empiquency in the signal as a function of time. The range of empiquencies contained in the IMFs varies over time but the first IMF always contains the locally highest one in the signal.

Example 5.2. We show the sifting process with a slightly more complicated signal defined by Equation 5.9 and plotted in Figure 5.5.

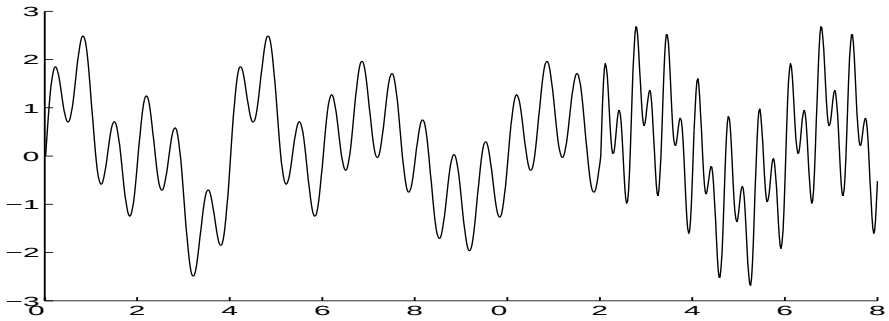


Figure 5.5. Plot of the test signal in Equation 5.9.

$$x(t) = \begin{cases} \sin(\pi t) + \sin(2\pi t) + \sin(6\pi t) & 0 < t \le 3 \\ \sin(\pi t) + \sin(6\pi t) & 3 < t \le 6 \\ \sin(\pi t) + \sin(6\pi t) + \sin(12\pi t) & 6 < t \le 9 \end{cases} \tag{5.9}$$

The signal has different frequency components in different parts. In the first two thirds of the signal the component of the highest frequency is $\sin(6\pi t)$, while in the last third of the signal the component of the highest frequency is $\sin(12\pi t)$. This is also the result of the sifting for the first IMF shown in Figure 5.6. The start of the sifting with first upper and lower envelope is shown in Figure 5.6 (a) and result from subsequent iterations is presented in Figure 5.6 (b-f).

The first IMF is shown in Figure 5.6 (f) and is a signal very similar to the signal given by Equation 5.10.

$$IMF1 = \begin{cases} \sin(6\pi t) & 0 < t \le 6 \\ \sin(12\pi t) & 6 < t \le 9 \end{cases} \tag{5.10}$$

The sifting for the second IMF is presented in Figure 5.7 in the same manner as the sifting for the first IMF is presented in Figure 5.6. The result should, in theory, give us the signal given by Equation 5.11 if we were concerned with only the inter frequency components in the signal. Due to the limitations of the implementation, further discussed in section 5.1.3, and the fact that different signal components influence each other and produce intra frequency effects in the signal, we have another result, shown in Figure 5.7.

$$IMF2 = \begin{cases} \sin(2\pi t) & 0 < t \le 3 \\ \sin(\pi t) & 3 < t \le 6 \\ \sin(6\pi t) & 6 < t \le 9 \end{cases} \tag{5.11}$$

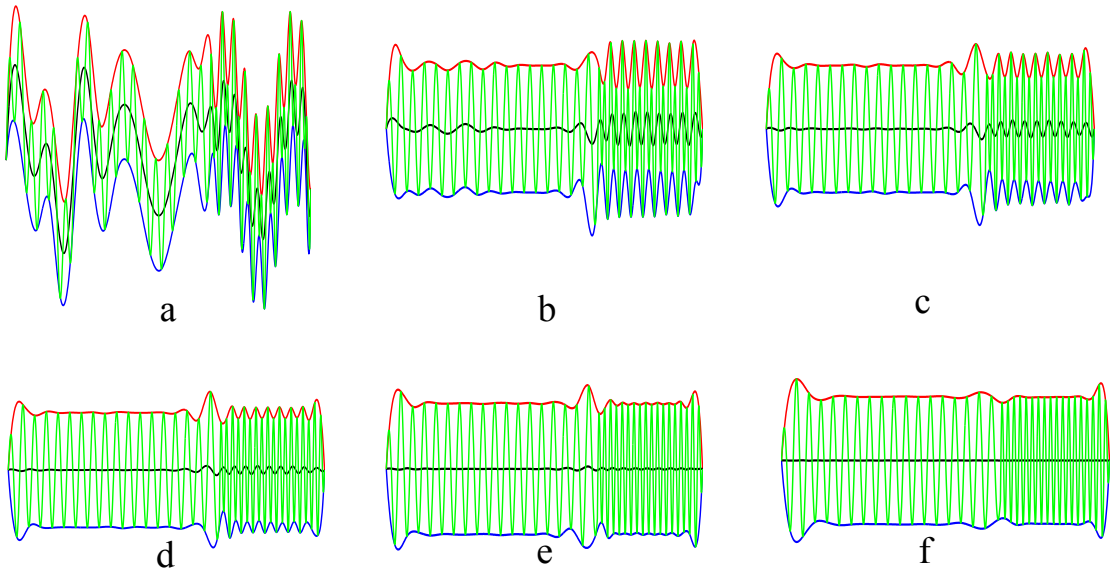


Figure 5.6. Iteration for the first IMF.

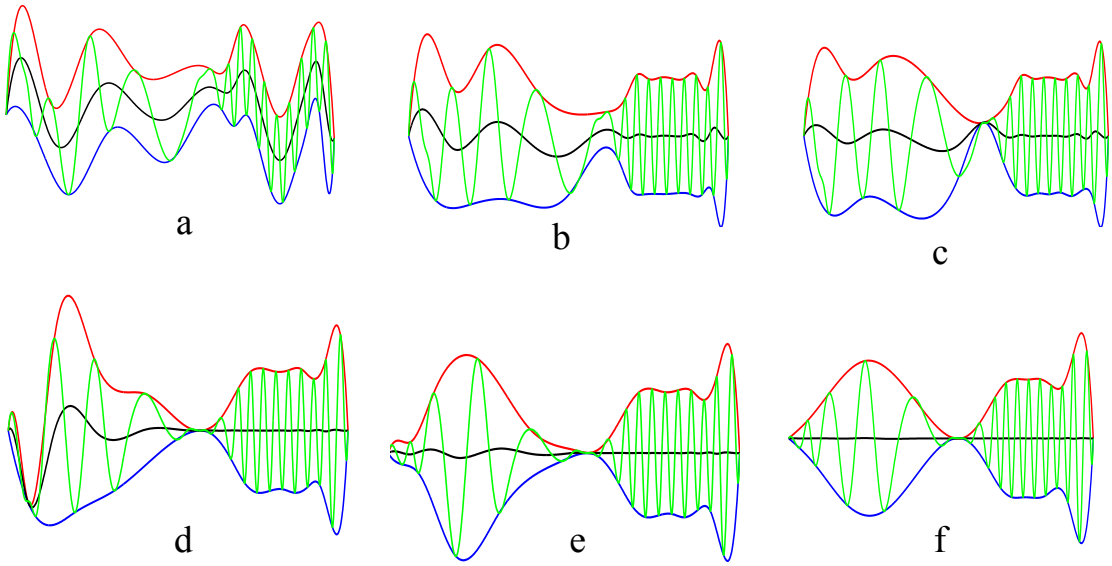


Figure 5.7. Iteration for the second IMF.

The lowest frequency component used in the composition of the test signal appears in the sifting for the third IMF, Figure 5.8, the next IMFs holds rest signals, Figure 5.9 to Figure 5.12, and the residue is a monotonic signal shown in Figure 5.13.

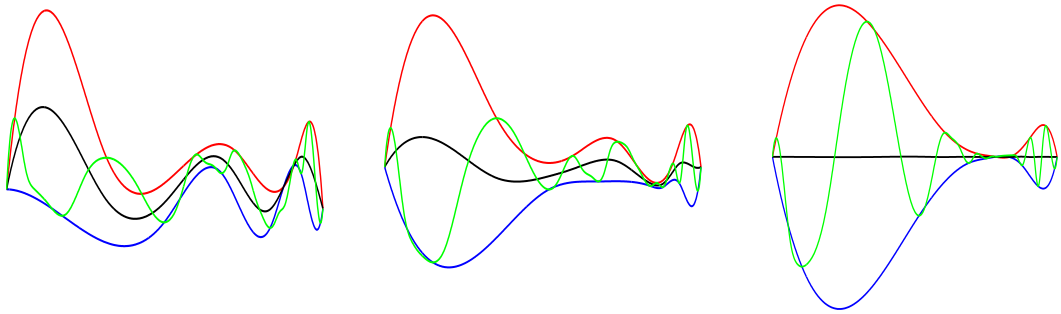


Figure 5.8. Iteration for the third IMF.

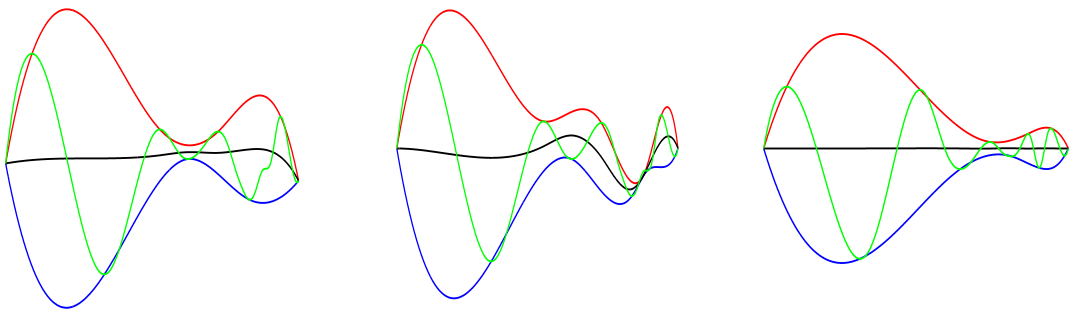


Figure 5.9. Iteration for the fourth IMF.

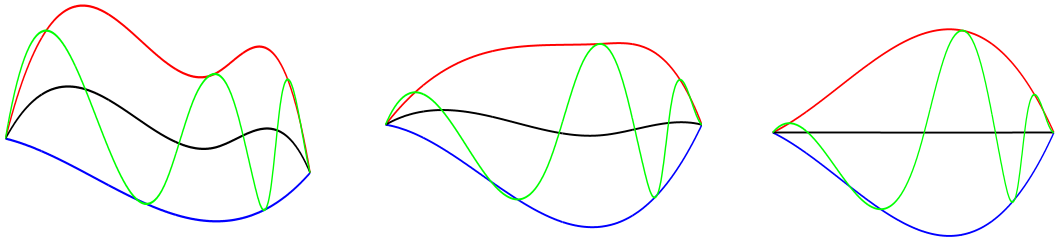


Figure 5.10. Iteration for the fifth IMF.

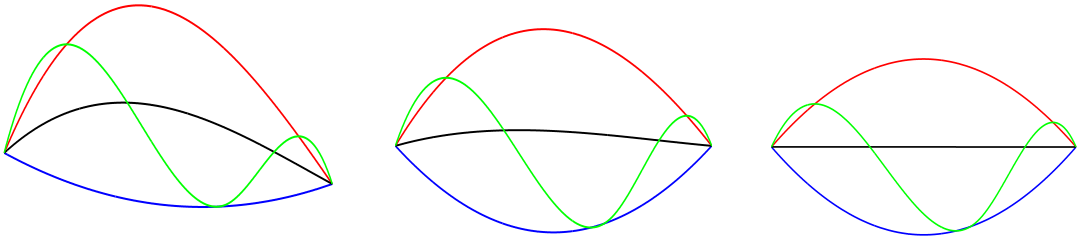


Figure 5.11. Iteration for the sixth IMF.

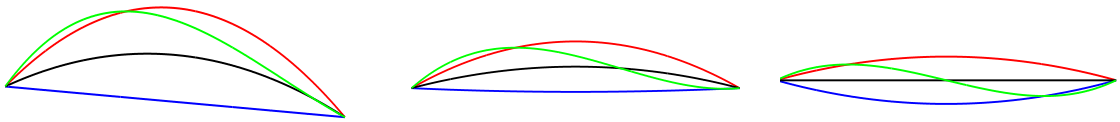


Figure 5.12. Iteration for the seventh IMF.

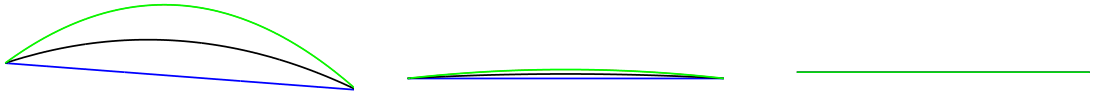


Figure 5.13. Iteration for the eighth IMF and the residue.

Example 5.3. When we use the composite chirp test signal from Example 5.1 in section 5.1.1, the result of the sifting for the first IMF is shown in Figure 5.14.

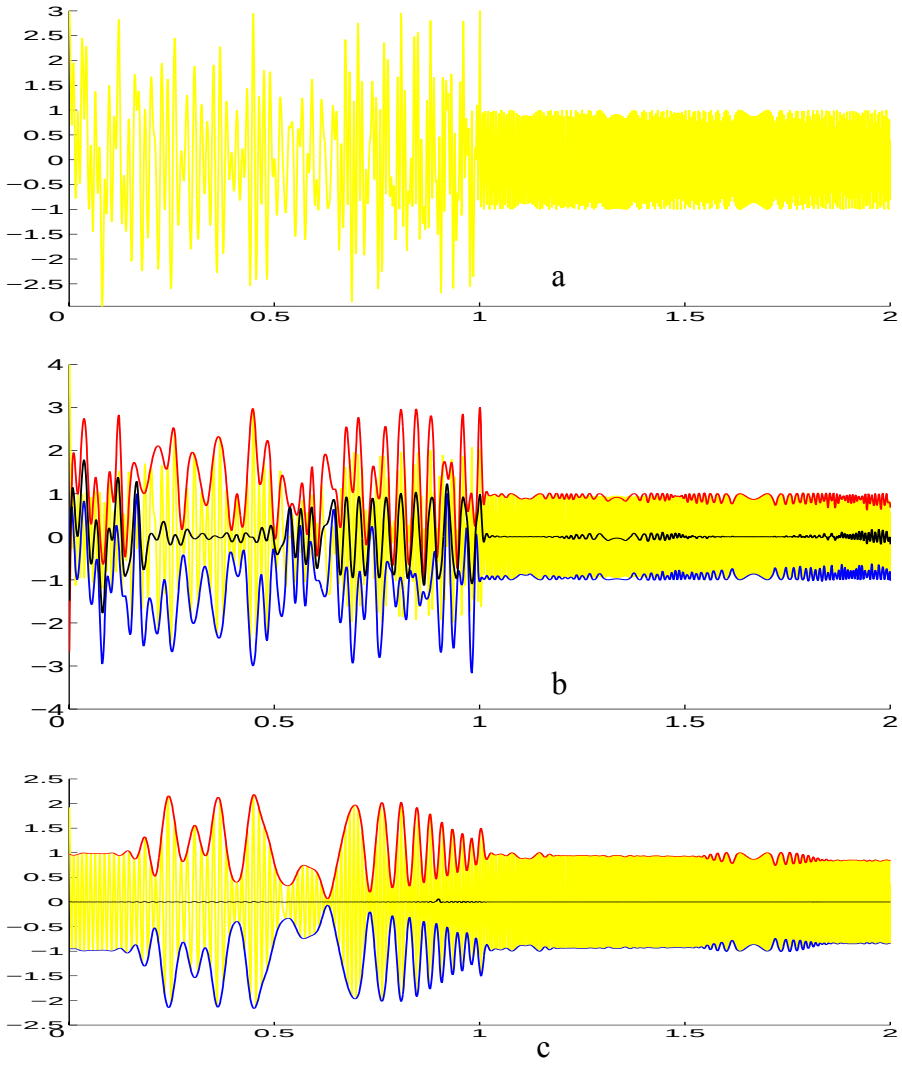


Figure 5.14. a) The composite chirp, b) the result of one iteration in the IMF sifting process with the upper and lower envelope and their mean plotted in the same figure, c) the resulting first IMF with the upper and lower envelope and their mean.

The composite chirp is shown in Figure 5.14 (a) and the result of one iteration in the IMF sifting process is shown in Figure 5.14 (b) in yellow with the upper envelope in red and the lower envelope in blue; their mean is shown in black. The resulting first IMF with the upper and lower envelope and their mean is shown in Figure 5.14 (c). In the latter case the envelopes are symmetric around their mean which is close to zero.

In Figure 5.15 the IMFs of the composite chirp signal are plotted together with their respective HHT. This shows the frequency separation capability of the EMD. The highest frequencies of the signal are represented by the first IMF while the next highest frequencies are represented by the second IMF etc. The EMD does not separate the signal into its original components.

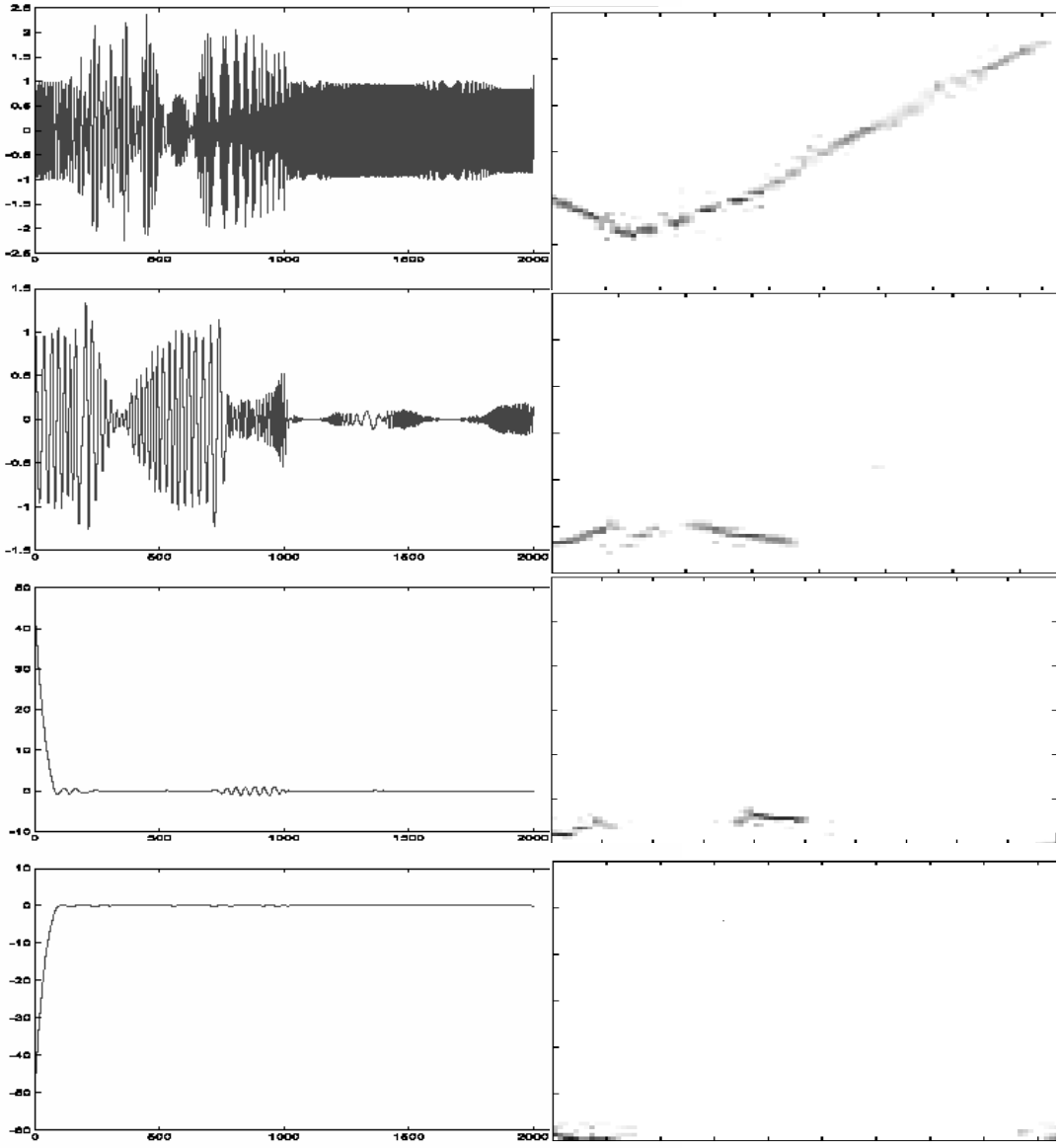


Figure 5.15. The first IMFs and the residue for the composite chirp (left) and their respective HHT (right).

5.1.3 Interpolation methods

The interpolation to produce envelopes from the extrema points can be done in different ways. The work done in [36] states that spline interpolation should be used. We here use piecewise cubic spline interpolation and piecewise cubic Hermite interpolation. These methods produce slightly different EMD results. The two interpolation methods are both splines but of different degrees.

On each subinterval $x(t)$, $k \le t \le k+1$, let *P(x)* be the interpolant to the given values and certain slopes at the two end points. Between any two neighbouring data sites *x(k)* and *x(k+1)*, *x(t)* is a polynomial. Neighbouring polynomials match in value, first, and second derivative, across their common data site.

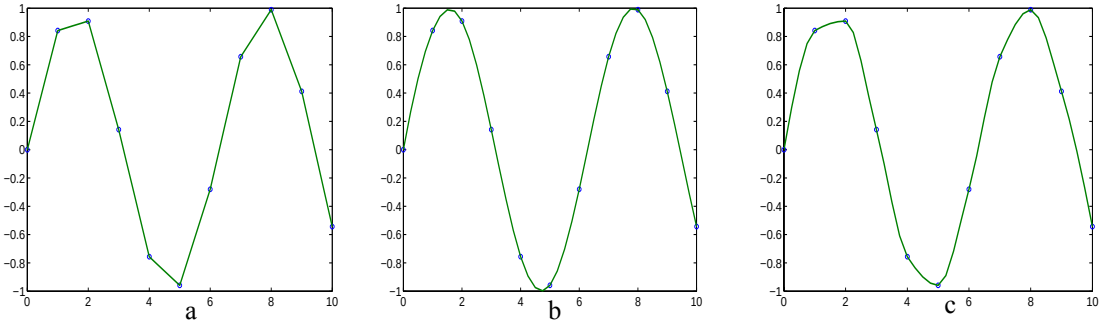


Figure 5.16. a) linear interpolation, b) piecewise cubic spline interpolation, c) piecewise cubic Hermite interpolation of regular data points.

For piecewise cubic Hermite interpolation [44], the interpolator is denoted $P_p(x)$. The first derivative, $P_p'(x)$, is continuous, but $P_p''(x)$ is not necessarily continuous. The slopes at the $x(k)$ are chosen in such a way that $P_p(x)$ is shape preserving and respects monotonicity. This means that, on intervals where the data are monotonic, so is $P_p(x)$, and at points where the data have a local extremum, so does $P_p(x)$: overshoots does not exist. The function $P_s(x)$ supplied by the piecewise cubic spline interpolation is constructed such that the slopes at the $x(k)$ are chosen to make $P_s''(x)$ continuous also. This makes $P_s(x)$ smoother and more accurate if the data are values of a smooth function. The effects of using linear interpolation, cubic Hermite interpolation, and cubic spline interpolation on regularly spaced data are shown in Figure 5.16. Note that in Figure 5.16 (b) the data points are no longer the extrema points of the interpolated signal. The interpolation has produced overshoots in order to achieve the continuous second derivative. These effects introduce new extrema points that may not have been present in the original signal or move the extrema point to another location.

The EMD involves the interpolation of irregularly spaced data. It is preferred that these stay as extrema points. Linear interpolation, cubic Hermite interpolation, and cubic spline interpolation on irregularly spaced data are shown in Figure 5.17. Only linear interpolation and cubic Hermite interpolation keep the extrema points in place. These effects are further treated in the examples.

Using the cubic spline interpolation method is good when decomposing signals, but at the same time it introduces new higher frequencies at false positions in longer periods of very low frequencies present in a surrounding of higher frequencies. This is shown in the Example 5.4 where the test signal from Example 5.2 is used, only modified such that the middle part is set to zero, Equation 5.12.

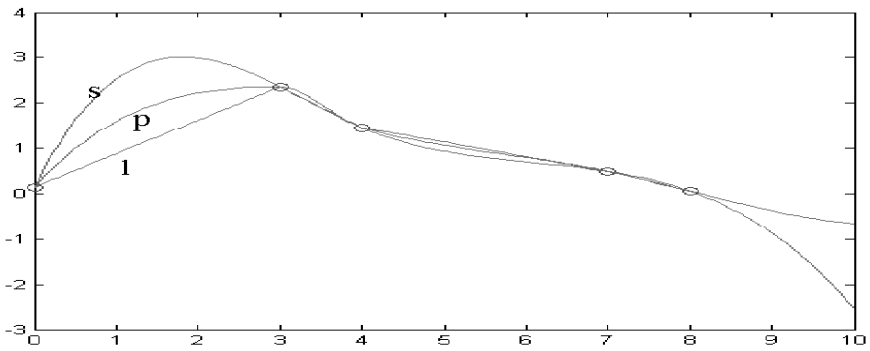


Figure 5.17. l = linear interpolation, s = piecewise cubic spline interpolation, p = piecewise cubic Hermite interpolation of scattered data points.

Example 5.4. The test signal is plotted in Figure 5.18. The result of using the cubic spline interpolation method is shown in Figure 5.19 in the sifting for the first IMF and Figure 5.20 in the sifting for the second IMF. The cubic Hermite interpolation does not introduce any new frequencies for the same case, shown in Figure 5.21 in the sifting for the first IMF and Figure 5.22 in the sifting for the second IMF, but to the cost of slower convergence of the sifting process and a larger amount of IMFs found for the same value on the stop criteria. When the cubic spline interpolation method is used the sifting introduces new frequency components in the zero zone, but the cubic spline interpolation method is recommended for most "normal" signals.

$$x(t) = \begin{cases} \sin(\pi t) + \sin(2\pi t) + \sin(6\pi t) & 0 < t \le 3 \\ 0 & 3 < t \le 6 \\ \sin(\pi t) + \sin(6\pi t) + \sin(12\pi t) & 6 < t \le 9 \end{cases} \tag{5.12}$$

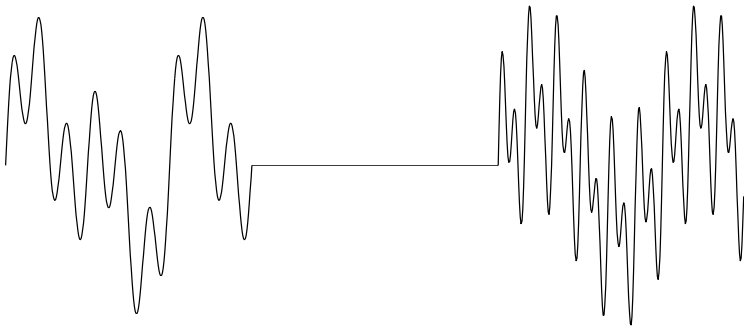


Figure 5.18. Test signal.

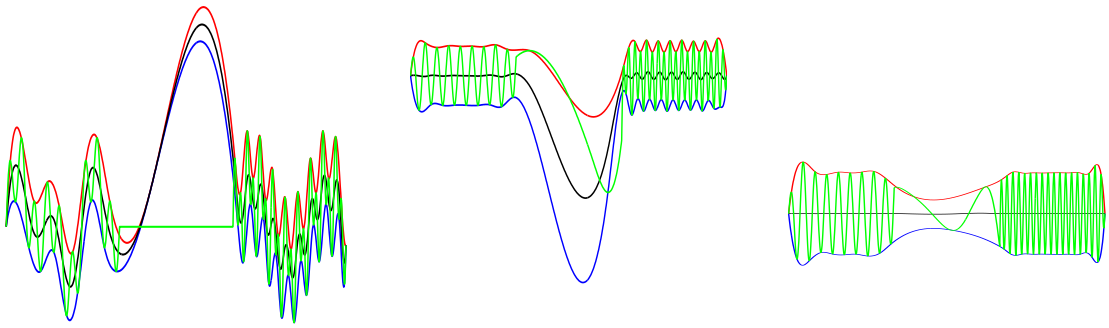


Figure 5.19. Iteration for first cubic IMF.

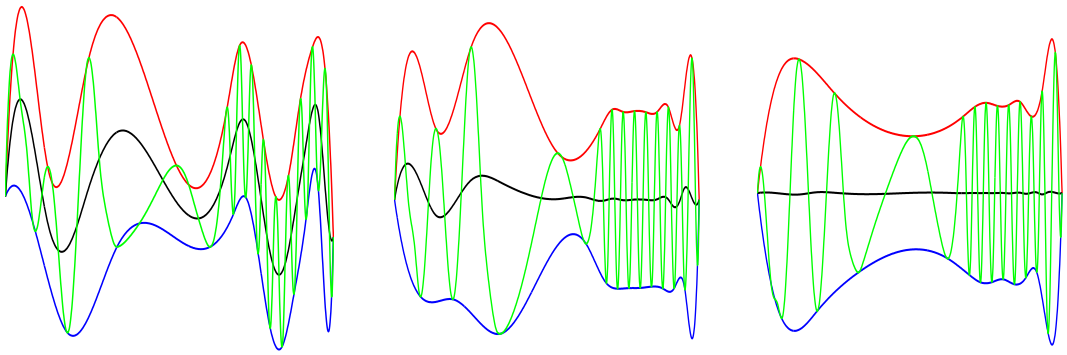


Figure 5.20. Iteration for second cubic IMF.

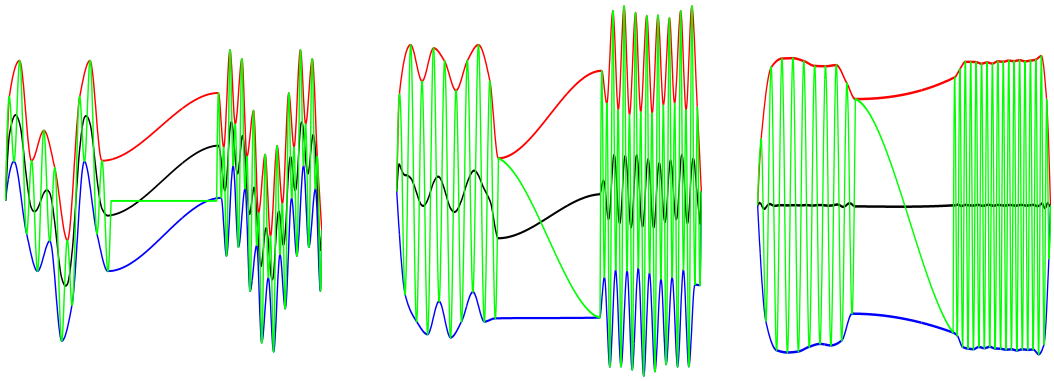


Figure 5.21. Iteration for first Hermitian IMF.

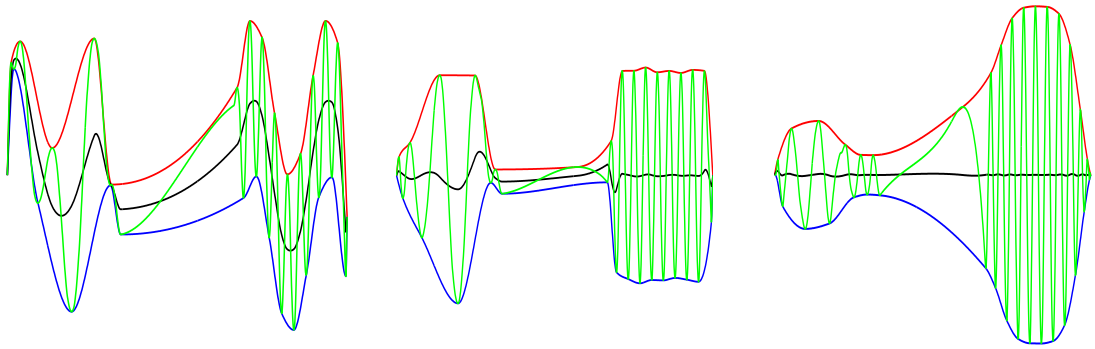


Figure 5.22. Iteration for second Hermitian IMF.

Example 5.5. To further examine the cubic spline interpolation and the cubic Hermite interpolation, we use a test signal composed of two different sinuoids. We apply both interpolation methods and look at the frequency separation capability of the EMD. The test signal

$$\sin(2\pi t) + \sin(6\pi t) \tag{5.13}$$

is plotted in Figure 5.23. The signal is simple and pure, both methods separate the first two IMF as signals that look similar to the input signal components. In Figure 5.24 (a) the signal component of the highest frequency is plotted, in Figure 5.24 (b) the first cubic Hermite interpolation IMF is plotted and in Figure 5.24 (c) their difference is plotted. The same is done for the second cubic Hermite interpolation IMF in Figure 5.25(b), which is compared to the signal component of the next highest frequency, plotted in Figure 5.25(a). Their difference is plotted in Figure 5.25 (c). The error is approximately 5% of the signal amplitude in the centre of the signal where the edge effects do not affect the signal.

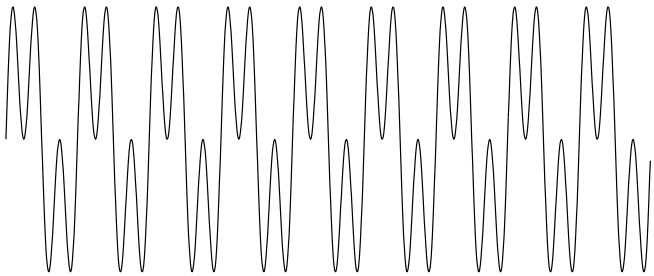


Figure 5.23. Test Signal

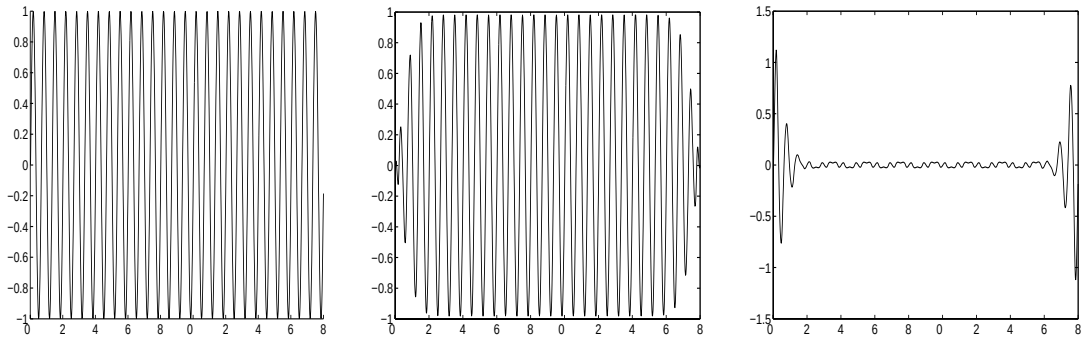


Figure 5.24. a) sin(6πt), b) first Hermitian IMF, c) Their difference.

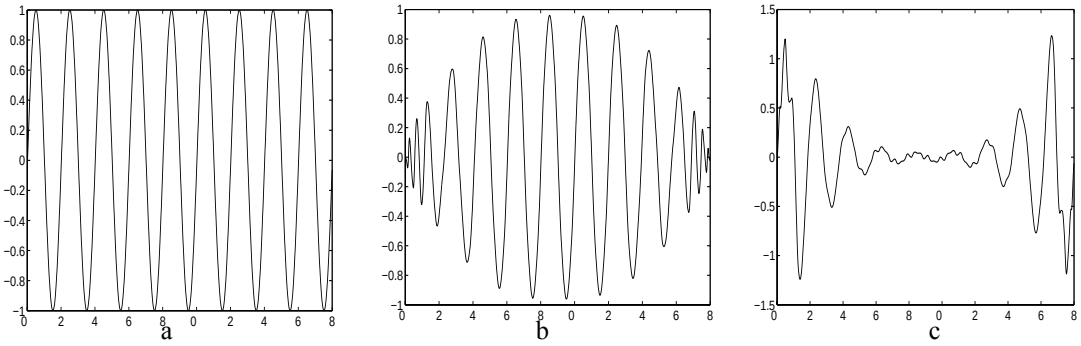


Figure 5.25. a) sin(2πt), b) second Hermitian IMF, c) Their difference.

When we do the same for the cubic spline method we have a better performance. In Figure 5.26 (a) the signal component of the highest frequency is plotted, in Figure 5.26 (b) the first cubic spline interpolation IMF is plotted and in Figure 5.26 (c) their difference is plotted. The same is done for the second cubic spline interpolation IMF in Figure 5.27 (b), which is compared to the signal component of the next highest frequency, plotted in Figure 5.27 (a). Their difference is plotted in Figure 5.27 (c). The error is approximately zero in the central region.

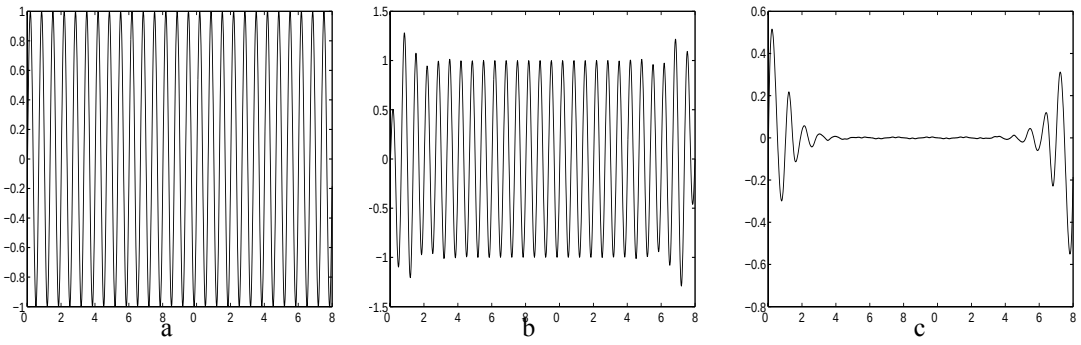


Figure 5.26. a) sin(6πt), b) first cubic IMF, c) Their difference.

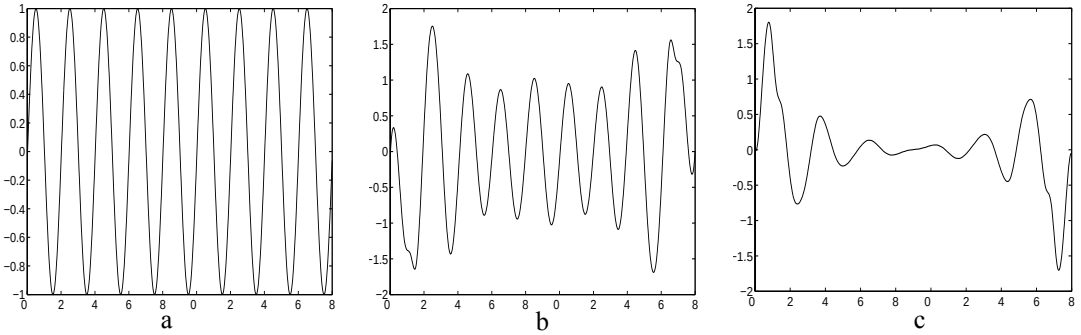


Figure 5.27. a) sin(2πt), b) second cubic IMF, c) Their difference

The cubic spline interpolation has worse behaviour at the edges than cubic Hermite interpolation due to the treatment of edge points. No effort is made on this subject in this work, some solutions to the edge problem for time signals can be found in [36].

5.1.4 Empiquency

The EMD is a truly empirical method, not based on the Fourier frequency approach but, as we will see, related to the locations of extrema points and zero crossings. Based on this we use the concept of *empiquency* [53], short for empirical mode frequency, instead of a traditional Fourier-based frequency measure to describe the signal oscillations. Let d be the distance between two neighbouring extrema points. In [36] the concept of *time scale* is presented. It can be seen as the mean of all d in the signal. In this context the empiquency is the *local time scale.* The measure of empiquency is defined as "One half the reciprocal distance between two consecutive extrema points":

$$f_e = \frac{1}{2d} \tag{5.14}$$

The empiquency value f_e is assigned to every position between the respective extrema points. Each extrema point influences more than one empiquency definition; we let it take the highest value of the two empiquency values it defines.

The empiquency is accompanied by an amplitude describing the signal at the actual position. At each extrema point the empiquency amplitude A_e is the absolute value of the extrema point. At all other positions, the empiquency amplitude takes the value of the absolute mean of the extrema points that define the empiquency for the actual position.

Thus,

$$A_e(p)) = \begin{cases} |a| & \text{if p is an extrema point} \\ \dfrac{|a| + |b|}{2} & \text{if p is not an extrema points} \end{cases} \tag{5.15}$$

where a and b are values of neighbouring extrema points.

The very special properties of the IMFs are that these are locally zero mean and ideally do not have more than one extrema point between two neighbouring zero crossings. Because of this relationship between zero-crossings and extrema points there is also a relation between the empiquency and the concept of *sequency*. Sequency is defined as "the average number of zero crossings per second divided by 2" [31]. For a sine or cosine the frequency and the sequency is the same, but the concept of sequency has a meaning for other signals such as Walsh functions or IMFs as well.

Example 5.6. Maximum empiquency within a signal segment is found by examining the space between the extrema points. Figure 5.28 and Figure 5.29 show two discrete time signals, both with the properties of an IMF but with different maximum empiquency, (0.5, in a normalized scale, for the signal in Figure 5.28 and 0.167 for the signal in Figure 5.29). Later on we will assume that a signal like the one in Figure 5.29 can be reconstructed with low distortion from a subsampled version, while a signal example like the one in Figure 5.28 needs every sample to be represented without distortion due to its high maximum empiquency.

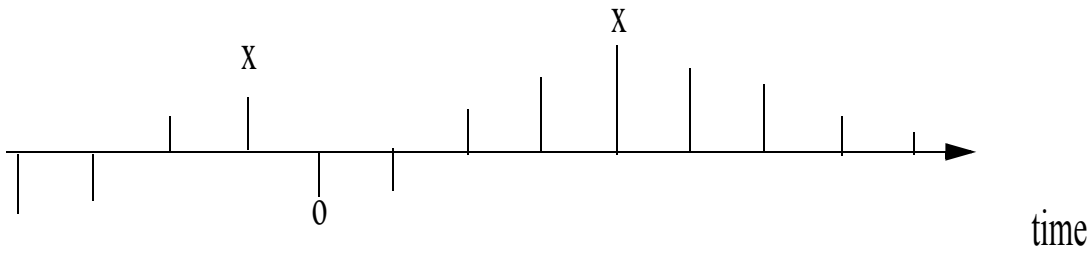


Figure 5.28. A discrete time signal with its three extrema points marked **x** for local maxima and **o** for local minima. Maximum empiquency 0.5.

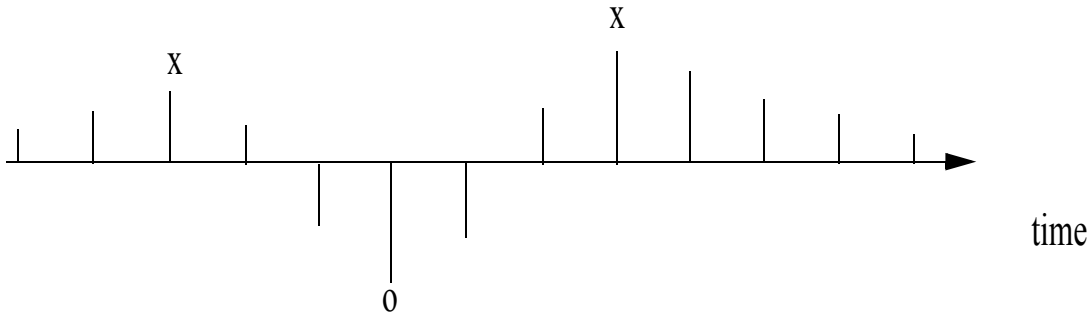
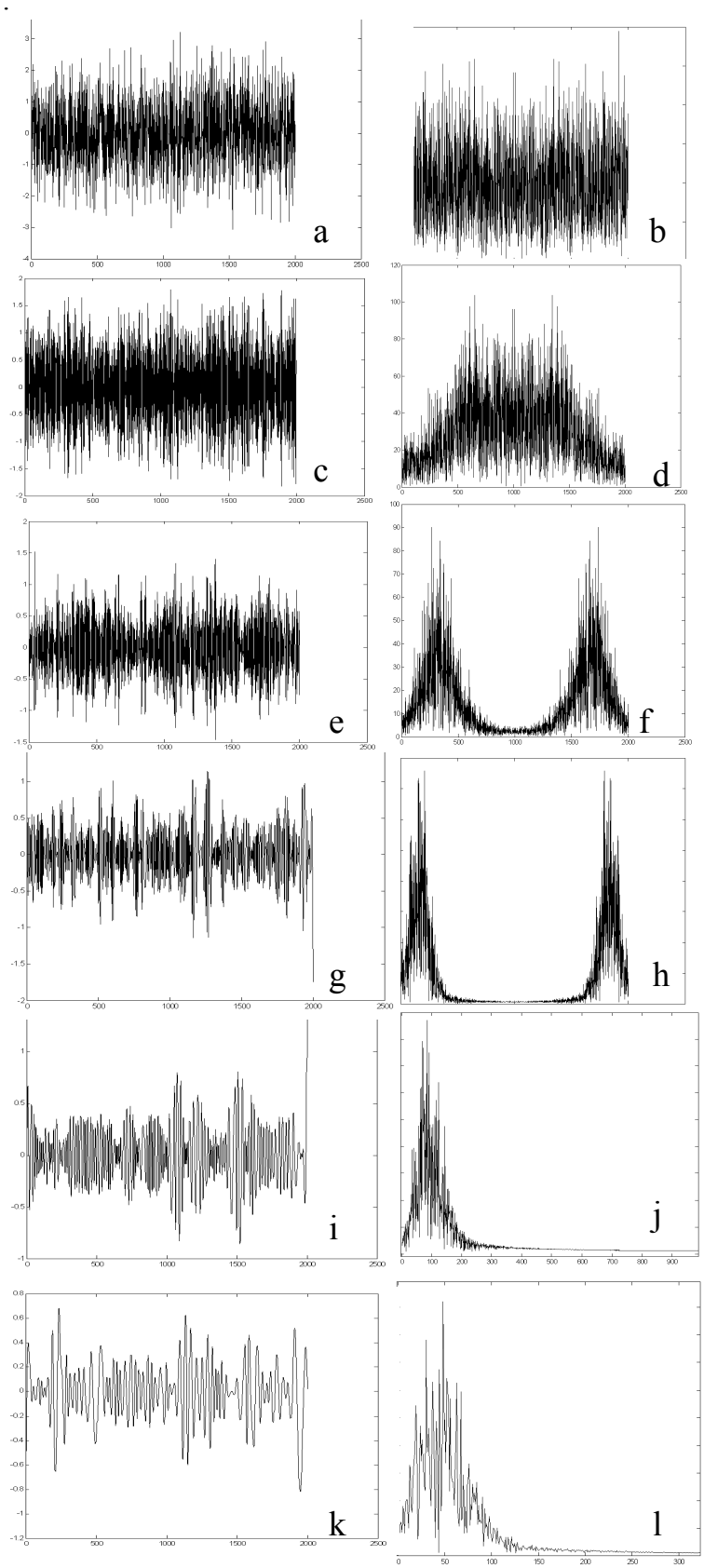


Figure 5.29. A discrete time signal with its three extrema points marked **x** for local maxima and **o** for local minima. Maximum empiquency 0.167.

5.1.5 EMD of noise

The behaviour of the EMD as a filter bank is highlighted in [26] through the analysis of noise. We review this work with an example using the cubic spline interpolation. The test signal used is white Gaussian noise with a variance of 1. The result from the EMD is shown in Figure 5.30 (a-y). The images in the left column show the time signal IMF and the middle column shows the fast Fourier transform (FFT) of the corresponding IMF. In Figure 5.30(b), (d), (f), (h), the spectrum density plot shows the whole result and thus the negative frequency components appear in the right half of the plot. The rest of the frequency plots show only positive frequency components in a zoomed view. The test signal is shown in Figure 5.30 (a) and its power spectra by means of the FFT of the signal is shown in Figure 5.30 (b). The first IMF show a “high-pass filtered” noise signal in Figure 5.30 (c) and Figure 5.30 (d). The rest of the IMFs show “band-bass filtered” noise signals with their centre frequency f_c decreasing in an octave band manner, just like a filter bank. This experiment shows that the EMD sorts the frequency content of a noise signal in the same way as it sorts any other signal. We should have this in mind when treating noisy signals. It is sometimes assumed that the noise of a signal is totally contained in the first IMF, an assumption that obviously, from this example, is not true.



zero mean Gaussian noise, variance1

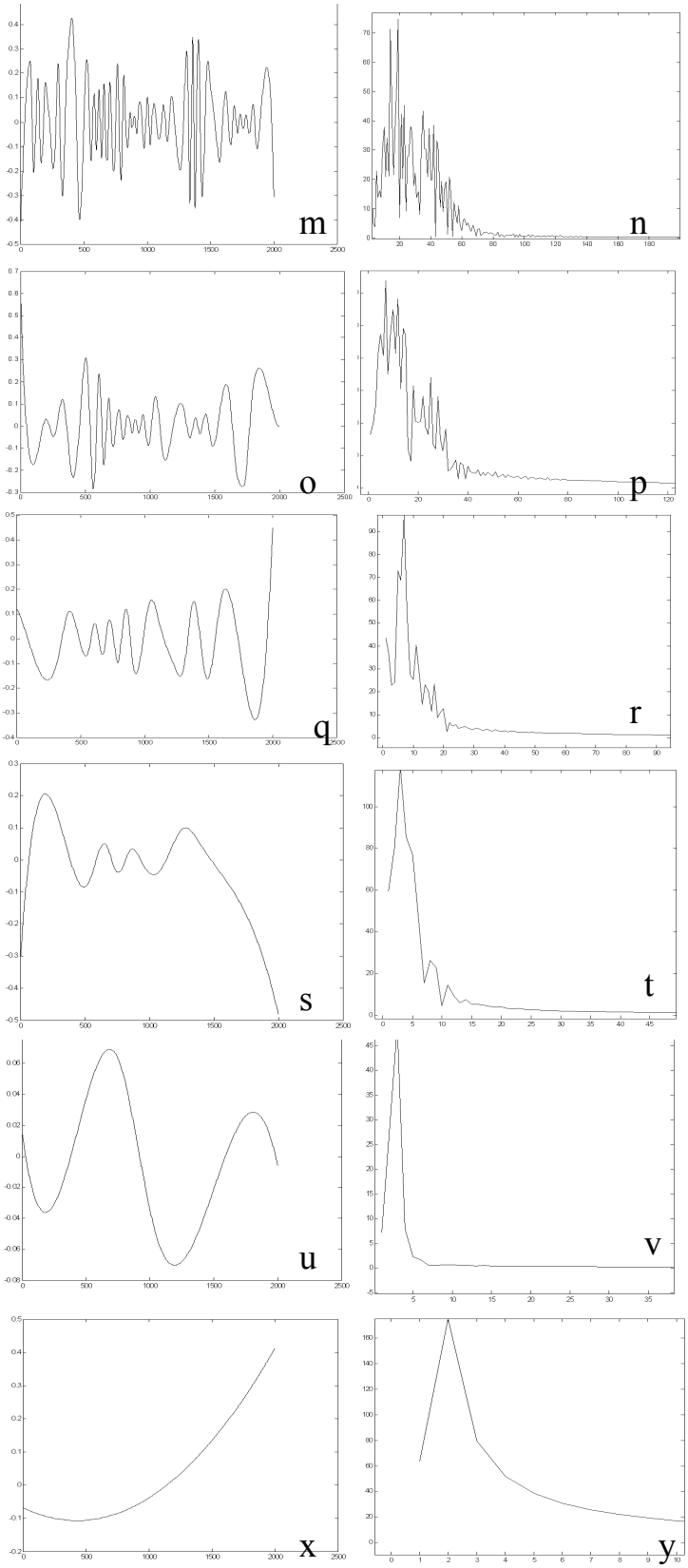
first IMF, variance 0.5750 centre frequency f_c=5000Hz

Second IMF variance 0.2117 centre frequency f_c=2500Hz

Third IMF variance 0.1225 centre frequency f_c=1250 Hz

Fourth IMF variance 0.0742 centre frequency f_c=500Hz

Fifth IMF variance 0.0545 centre frequency f_c=250Hz



Sixth IMF
var(c6)=0.0230
f_c=160Hz

Seventh IMF
var(c7)=0.0181
f_c=80Hz

Eighth IMF
var(c8)=0.0158
f_c=40Hz

Nineth IMF
var(c9)=0.0189
f_c=20Hz

Tenth IMF
var(c10)=0.0016
f_c=15Hz

Tenth residue
var(res10)=0.0233
f_c=10Hz

Figure 5.30. EMD of gaussian nose. Time plot of the IMF to the left, the spectral density from FFT of the IMF in the middle and specifications for the IMF to the right.

5.1.6 Conclusion

The concept of empiquency is new and is useful to describe time signals and, as will be seen later in this thesis, also when EMD is applied to images. HHT is suitable for time-frequency analysis. The performance depends on the interpolation method and its implementation. Cubic spline interpolation is probably the best interpolation method for EMD of time signals despite its drawbacks. This is an area where more work is needed, but this is not the focus of this thesis. The number of IMFs achieved with the EMD depends also on the choice of stop criterion value. If it is too large the number of IMFs is not large enough to make the empiquency separation complete. If it is too small the empiquency separation completes but computation time is longer than necessary.

5.2 EMD of two-dimensional signals

HHT, the EMD-based time-frequency analysis, presented in section 5.1.1, is only one of many applications made possible by the use of EMD. In the two-dimensional case, where adaptive spatial frequency separation presents a challenge, the EMD enables many new approaches for image processing applications.

The results and ideas in time domain applications using the EMD technique apply to two-dimensional signals, such as images, as well. Similarly to the one-dimensional EMD, the first IMF extracts the locally highest spatial empiquencies in the image while the second IMF holds the locally next highest spatial empiquencies, etc. The EMD in two dimensions provides a tool for image processing by its special ability to locally separate spatial frequencies that build texture. Assume that a texture consists of a superposition of a number of spatial frequency patterns. At each spatial position the texture is a result of the superposition of different spatial frequencies. The EMD sorts the spatial frequency components into a set of IMFs where the highest spatial frequency component of each spatial position is in the first IMF and the next highest spatial frequency component of each spatial position is in the second IMF, etc.

The use of the Hilbert transform for the creation of analytic signals in two dimensions is made possible by the introduction of a direction of reference in the Fourier domain [30], but will not be further treated in this thesis. In the two-dimensional EMD, the symmetry criterion on the IMF envelopes is relaxed. The stop criterion is based on the condition that the IMF envelope mean must be close enough to zero. This guarantees that we will find the IMF without actually checking for symmetric envelopes.

Other work on EMD in two dimensions typically generates a residue with many extrema points. In this chapter we propose an improved method to decompose an image into a number of IMFs and a residue image with a minimum number of extrema points.

The extension of the EMD to two dimensions relies on proper spline interpolation in two dimensions. The edge constraints are important, as the errors introduced there traverse into the image. The IMFs can be seen as spatial frequency subbands, with various centre frequency and bandwidth along the signal. Within each IMF both high and low empiquencies can coexist at different locations. The key word still is *locally*.

5.2.1 Sifting for the two-dimensional IMF

The sifting process for the IMF is here extended to two dimensional signals.

To find the first IMF, start with the image itself as input signal $in_{11}(m, n) = x(m, n)$. The first index is the IMF number, l=1..L, and the second index is the iteration number, k=1..K, in the sifting process. m and n represent the two spatial dimensions. To find the next IMF, use the residue corresponding to the previously found IMF as input signal $in_{21}(m, n) = r_1(m, n)$.

The sifting process to find the IMFs of a signal *x(m,n)*, comprises the following steps:

(1) Find the positions and amplitudes of all local maxima, and find the positions and amplitudes of all local minima in the input signal $in_{lk}(m, n)$.

(2) Create the upper envelope by spline interpolation of the local maxima and the lower envelope by spline interpolation of the local minima. Denote the envelopes $e_{max}(m, n)$ and $e_{min}(m, n)$. respectively.

(3) For each position *(m,n)*, calculate the mean of the upper envelope and the lower envelope.

$$em_{lk}(m, n) = \frac{e_{max}(m, n) + e_{min}(m, n)}{2} \tag{5.16}$$

The signal $em_{lk}(m, n)$ is referred to as the envelope mean.

(4) Subtract the envelope mean signal from the input signal

$$h_{lk}(m, n) = in_{lk}(m, n) - em_{lk}(m, n) \tag{5.17}$$

This is one iteration of the sifting process. The next step is to check if the signal $h_{lk}(m, n)$ from step 4 is an IMF or not. The process stops when the envelope mean signal is close enough to zero.

$$\left|em_{lk}(m, n)\right| < \varepsilon \qquad \forall(m, n) \tag{5.18}$$

In the same way as for time signals, forcing the envelope mean to zero will give us the wanted symmetry of the envelope and the correct relation between the number of zero crossings and the number of extremes that define the IMF. This way we will find the IMF without actually having to check for symmetric envelopes.

(5) Check if the mean signal is close enough to zero, based upon the stop criterion. If not, repeat the process from step 1 with the resulting signal from step 4 as the input signal a sufficient number of times.

$$in_{l(k+1)}(m, n) = h_{lk}(m, n) \tag{5.19}$$

When the stop criterion is met, k=K, the IMF is defined as the last result of (4).

$$c_l(m, n) = h_{lK}(m, n) \tag{5.20}$$

After the IMF $c_l(m, n)$ is found, define the residue $r_l(m, n)$ as

$$r_l(m,n) = in_{l1}(m,n) - c_l(m,n) \tag{5.21}$$

(6) The next IMF is found by starting over from step 1, now with the residue as the input signal.

$$in_{(l+1)1}(m,n) = r_l(m,n) \tag{5.22}$$

Steps (1) to (6) can be repeated for all the subsequent r_j. The EMD is completed when the residue, ideally, does not contain any extrema points. This means that it is either a constant or a monotonic function. The signal can be expressed as the sum of IMFs and the last residue

$$x(m,n) = r_L(m,n) + \sum_{j=1}^{L} c_j(m,n) \tag{5.23}$$

5.2.2 Implementation for image EMD

The EMD of images relies on proper spline interpolation in two dimensions. Some efforts to implement the EMD in two dimensions have been published [51,62] but the methods have not proven to be able to fully decompose the two-dimensional signal down to a residue with a low number of extrema points. We will give an example of such an implementation and explain its drawbacks. The residue signal should have only a few extrema points. Huang et al. [36] state that there should be no extrema points in the one-dimensional case and ideally we would like to have a similar criterion for the two-dimensional case.

It is mainly steps (1) and (2) in the sifting process that cause problems in the implementation of the EMD in two dimensions. We here present our solution to these problems. In step (1) we have the problem of selecting the extrema points. In 2D, there are many possibilities to define extrema, each one yielding a different decomposition. Since we are only concerned with discrete two-dimensional signals we simply extract the extrema points by comparing the candidate data point with its nearest 8-connected neighbours. This does not allow for saddle points to be considered as extrema points. In the case where saddle points are considered to be extrema points in the algorithm, these are both maxima and minima at the same point. This nails the saddle point to the zero mean level in the first sifting round. Clearly, more sophisticated methods to extract extrema points could handle the situation with saddle points to be considered as extrema points as well, but the extrema points defined by an 8-connected neighbourhood serves the purpose for EMD at this stage, with further improvement being possible. In the following, saddle points are thus not considered to be extrema points.

The border constraints are even more important in two dimensions than they are in the one-dimensional EMD case. One of the main objections for using spline interpolation in the EMD for two-dimensional signals such as images is that the borders cause too many problems. The set of extrema points is very sparse and since the interpolation methods only interpolate between points, the borders need special care. We propose the trick

of adding extra data points at the borders to the set of extrema points. These extra points are placed at the corners of the image and equally spaced around the border. Without these extra points, the areas not covered by the interpolation traverse into the image in the sifting process.

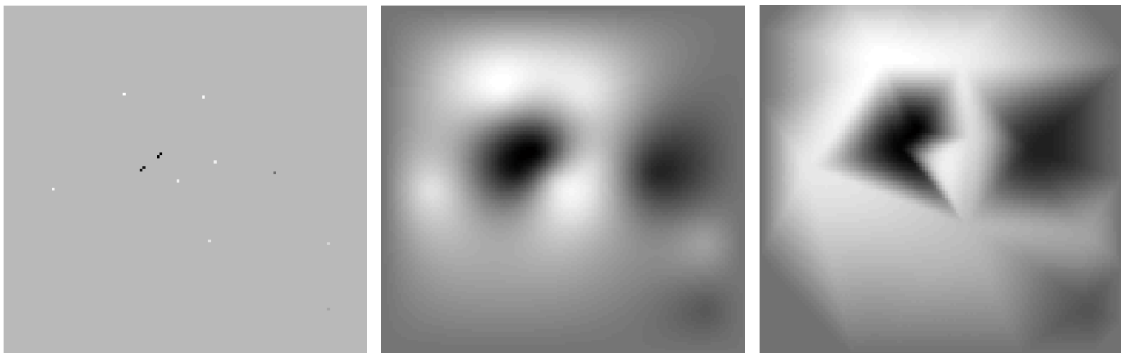
In the second step in the sifting process the problem is to fit a surface to the two-dimensional scattered data points representing the extrema points. The interpolated surface must go through each data point. Overshot must be avoided and the second derivative must be continuous everywhere for the signal to be smooth enough for two-dimensional EMD. The interpolation methods considered here are triangle-based cubic spline interpolation and thin-plate smoothing spline interpolation. The first is an example of an interpolation method using a piece-wise approach to interpolate the surface. It produces piecewise smooth surfaces and is based on Delaunay triangulation [67] of the data. The piecewise approach causes more problems in the two-dimensional case than with one-dimensional signals. Even though the interpolator produces continuous second derivatives, such as the cubic spline in [51] or the radial basis functions used in [62], the borders of the neighbouring pieces cause problems. It is possible that the piece-wise approach will work if we use another set of extrema points where the saddle points are included. This is not examined in this thesis but is left to future work.

We suggest that the thin-plate smoothing spline interpolation is used for the implementation of the two-dimensional EMD. This method gives a surface with continuous second derivative everywhere. The thin-plate smoothing spline algorithm [8] calculates the function f that minimizes the integral bending norm I_f, in Equation 5.24, for given scattered data in the plane. The integral is taken over the entire image, and involves the second derivatives of f.

$$I_f = \int\int_{R^2} \left(\left(\frac{\partial^2 f}{\partial x^2} \right)^2 + 2 \left(\frac{\partial^2 f}{\partial x \partial y} \right)^2 + \left(\frac{\partial^2 f}{\partial y^2} \right)^2 \right) \partial x \partial y \tag{5.24}$$

This means that the determination of the smoothing spline involves the solution of a linear system with as many unknowns as there are data points. This method turns out to successfully decompose an image into its IMFs and a smooth residue with no or only a few extrema points. We use the thin-plate smoothing spline interpolation for the two-dimensional EMD throughout this thesis.

Example 5.7. The scattered extrema points from the fourth IMF in Figure 5.35(d) are shown in Figure 5.31(a). Local maxima are plotted as light pixels and local minima are plotted as dark pixels. The different surfaces achieved when interpolating the same scattered data points using thin-plate interpolation and triangle-based cubic interpolation are shown in Figure 5.31(b,c). The artifacts when using a piece wise approach on very sparse data are caused by the sharp edges and the un-smooth borders of the different areas created by the triangulation. This can be seen in Figure 5.31(c). These sharp borders introduce new extrema points in the iteration for the next IMF and prevent the decomposition from finding and separating any lower spatial frequencies. This also forces the EMD method into producing IMFs containing features that were not in the image originally.



a b c

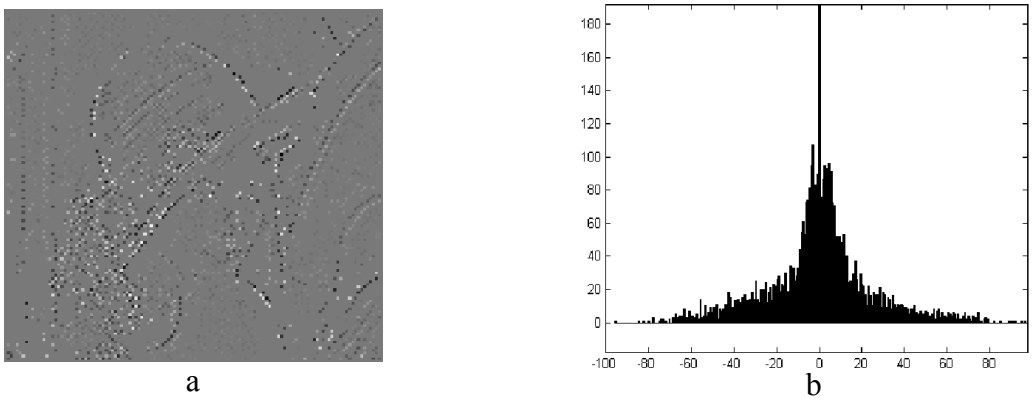
Figure 5.31. The different resulting surfaces when interpolating the scattered data points (a), using thin plate interpolation (b), and triangle-based cubic interpolation (c).

5.2.3 Significant extrema points

In the first and second IMFs the number of extrema points usually is very large. Not all of these are essential for the signal analysis We can reduce the number of extrema points by selecting the *significant extrema points* of the IMF. This is done by setting the extrema points with low amplitude to zero, as described by Equation 5.25.

$$b(m,n) = \begin{cases} 0 & if \quad |b(m,n)| \le T \\ b(m,n) & if \quad |b(m,n)| > T \end{cases} \tag{5.25}$$

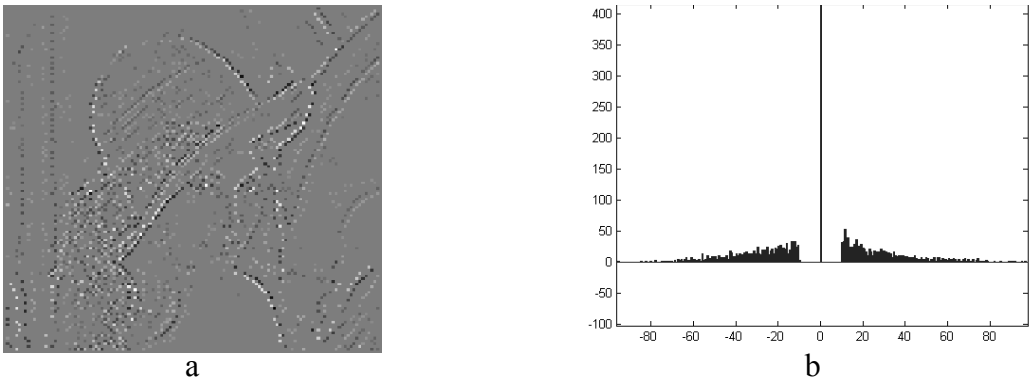
Figure 5.32(a) shows the extrema points of the first IMF at their position in the image where the rest of the pixels are set to zero. Figure 5.32(b) shows the corresponding histogram of the image in Figure 5.32(a).



a b

Figure 5.32. a) The extrema points of the first IMF at their position in the image, the rest of the pixels zeroed. b) histogram of the image in a.

If we let the coefficients with absolute amplitude lower than a suitable threshold, in this case 10, be set to zero we get the significant extrema points with respect to the threshold, shown in Figure 5.33.



a b

Figure 5.33. a) The image in Figure 5.32(a) with the smallest coefficients set to zero, b) histogram of the image in (a).

Keeping only the significant extrema points reduces the number of samples needed to represent the IMF, with only a minor effect in the reconstructed signal. However, errors will be introduced which will show up in the SNR measure.

This operation also serves as a tool for *noise reduction*. The insignificant extrema points are defined to be noise. It is sometimes assumed that the first IMF contains all the noise in the signal and that the noise can be removed by just skipping the first IMF. This is not true. Noise usually contains all empiquencies, both higher and lower than the maximum empiquency of the signal. The effects of EMD of noise was treated in section 5.1.5. Image processing is often performed on digitized photos. This means that any image is noisy, due to the physics of image capturing. High quality photos, such as the Lenna image, have a low noise level, not visible to the human eye, with a signal-to-noise ratio higher than 40 dB. For signals with moderate signal-to-noise ratio, we can still assume that the noise is of lower amplitude than the significant extrema points. We can then reduce the noise by setting the extrema points with low amplitude to zero, as described by Equation 5.25.

5.2.4 Image EMD

In this section we use three examples to illustrate the application of our method for image EMD calculations. The first example decomposes the 128x128 pixel Lenna image shown in Figure 5.34. The second example uses a 64x64 pixel detail part of the 512x512 pixel Lenna image and the third example treats a 64x64 detail part of the Barbara image in Figure 5.38.

Example 5.8. As an example of an image EMD we look at the well-known Lenna image, shown in Figure 5.34. The image's four IMFs and their corresponding residues are shown in Figure 5.35. This example clearly demonstrates the “band-pass filtering” effects of the EMD. The first IMF in a) and the first residue in e) can be seen as the result of a high-pass and low-pass filtering process. Adding these together reconstructs the original. For creation of the second IMF the first residue in Figure 5.35(e) is treated and the result shows as the second IMF in b) and the second residue in Figure 5.35(f). Using these to reconstruct the original we add a), b), and f) according to Equation 5.23. Continuing the decomposition gives four IMFs and a final residue with only a few extrema points.

Figure 5.34. The Lenna image at 128x128 pixel size.

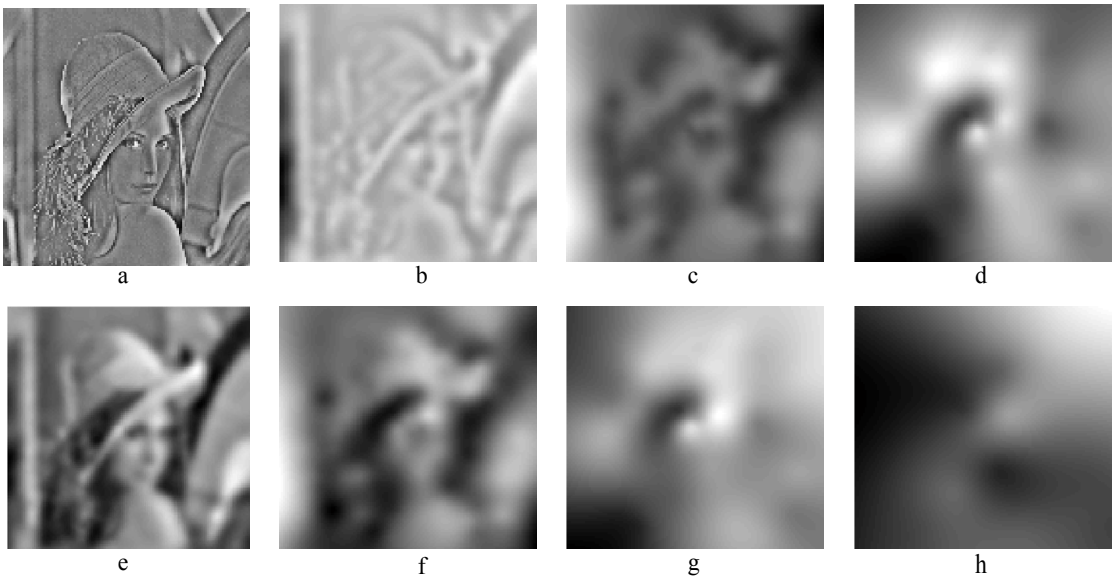


Figure 5.35. a) First IMF, b) second IMF, c) third IMF, d) fourth IMF, e) first residue, f) second residue, g) third residue, h) fourth residue.

Example 5.9. Figure 5.36 shows a part of the textured hat of the Lenna image. This texture is only visible in the high resolution 512x512 pixel version of the image. This example shows the EMD of a highly detailed texture image in Figure 5.37, using a medium level of the stop criterion. The first IMF captures the texture component with high spatial frequency of the hat and in the same IMF the high frequency noise in the background. The texture components of lower spatial frequency remain in the first residue, as do the shape and curvature of the hat. Next step is to separate the first residue into the second IMF and second residue. Here the second IMF still holds some high spatial frequencies and the second residue holds the shape and curvature of the hat. While decomposing this image down to a residue with only a few extrema points, we see the potential of EMD to evolve into a texture representation tool. This application is not treated in this thesis but left to future work.

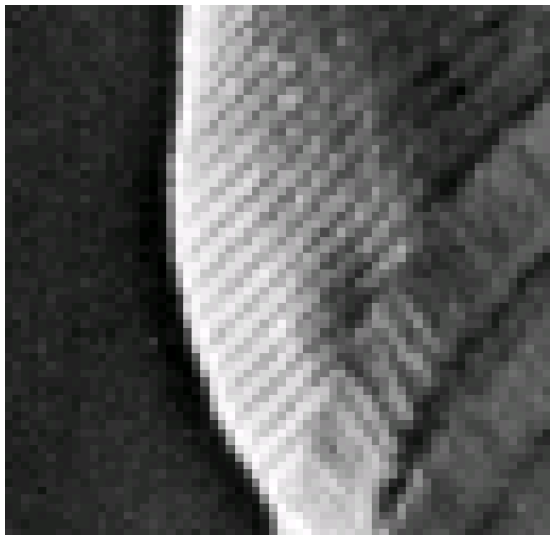


Figure 5.36. Part of the hat of Lenna image, 512x512.

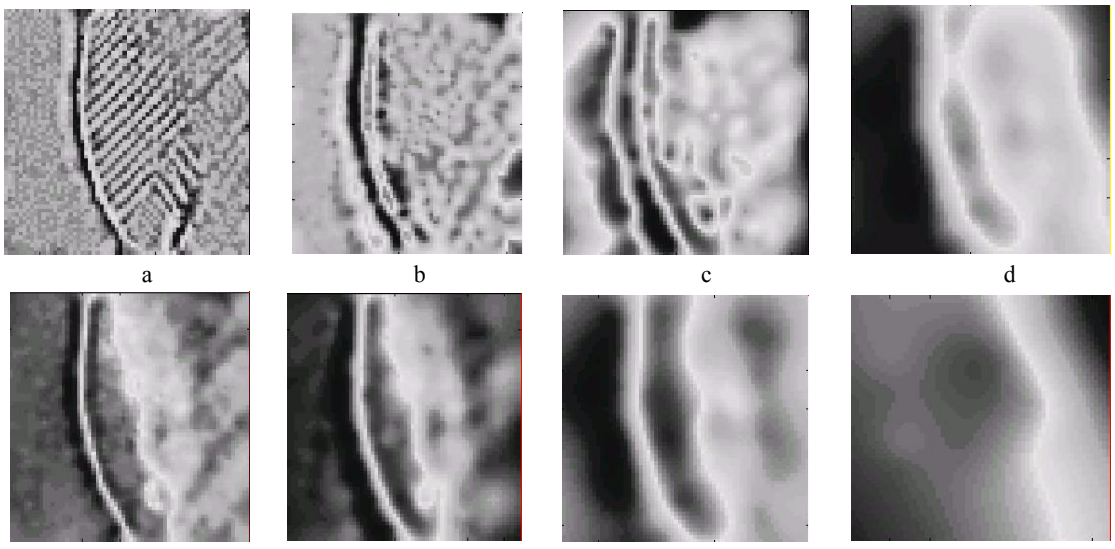


Figure 5.37. a) First IMF, b) second IMF, c) third IMF, d) fourth IMF, e) first residue, f) second residue, g) third residue, h) fourth residue.

Example 5.10. Figure 5.38 shows the Barbara image, with the lower right corner cut as test image. This image contains many parts with high spatial frequency and is often used to demonstrate the performance of image coders and texture analysis methods. We use the lower right corner of the image for this example. The EMD is shown in Figure 5.39, with, from left to right: the first IMF, the second IMF, the third IMF and the third residue. This is an example of the use of too large a value of the stop criterion. With a smaller value the first two IMFs would have been separated into some more IMFs and more information of the image would have been revealed.

Figure 5.38. Left: Barbara. Right: lower right corner zoomed.

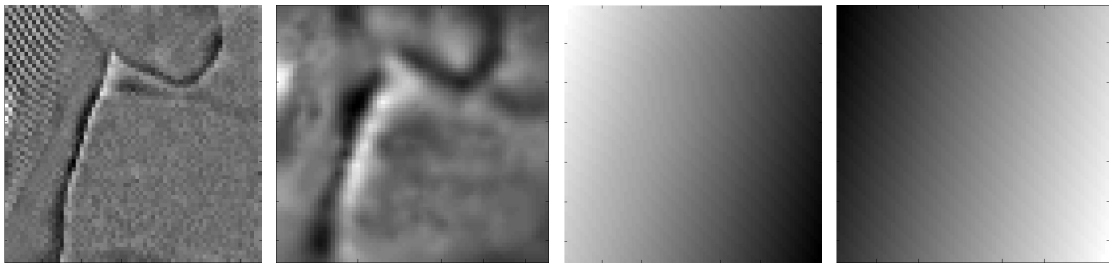


Figure 5.39. From left to right, the three IMF s of the image to the right in Figure 5.38, and the residue

It should be noted that all these IMFs are approximately zero mean, while the DC level of the signal is contained in the residue. These IMFs have the very special property of having one extrema point between two zero-crossings in almost any direction, except where there are saddle points. This will be used in the next chapter to get a more efficient representation of the EMD.

5.2.5 Conclusion

The EMD concept is successfully extended into two dimensions. The main problem with the implementation is the detection of the extrema points and the interpolation of the scattered extrema points. Performance of the EMD depends on the interpolation method and its implementation.

5.3 Summary

In this chapter the Empirical Mode Decomposition (EMD) was introduced. The use of EMD in the Hilbert-Huang Transform (HHT) was viewed as an example of EMD for time signals. Different interpolation methods were discussed for time signals. The concept of empiquency is introduced for further use in subsequent chapters.

The EMD was extended to two dimensions and we proposed an improved method to decompose a full image into a number of IMFs and a residue image with a minimum number of extrema points. The selection of significant extrema points was discussed as a tool for noise reduction.

Chapter 6

Compression of the EMD

In Chapter 5 the EMD for images was developed. In this and the following chapters we will investigate how to use it for compression. In this chapter a number of methods to compress the EMD are tested. The set of IMFs and the residue image is a very redundant way to represent an image. Our first goal is to represent the image by its EMD with the same number of bits as the original image. Our second goal is to invent an image coder that achieves reasonable compression. Throughout this chapter the test image used is the part of Lenna 512 shown in Figure 5.36 called "detail one".

First we use a memory-less coding strategy, an entropy coder, on each IMF and the residue to test the compression potential of the EMD.

The next attempt to use the EMD for compression is the Extrema Point Coder [51]. This method is examined here both for time signals and images. The EMD sorts the empiquencies in such a way that there is only one empiquency present at each position for each IMF. Although the empiquency changes with position in the IMF the hypothesis for the next compression method is that within a reasonable small image block the empiquency should be approximately the same. This way the block-based DCT could represent the image with only one or two of the DCT coefficients for each block.

Disregarding the blocking, the DCT of the IMFs shows an interesting structure. Apart from the first IMF they have all significant components in the low-frequency region of the coefficient matrix. Zeroing out everything else gives an efficient representation of the IMF and the compression result is good. But there still remains to find a way to compress the first IMF.

6.1 Entropy coding of the EMD

The scheme for entropy coding of the EMD is presented in Figure 6.1. The image is decomposed into its IMFs and a residue. These are entropy coded by a Huffman coder applied to the quantized IMFs and residue.

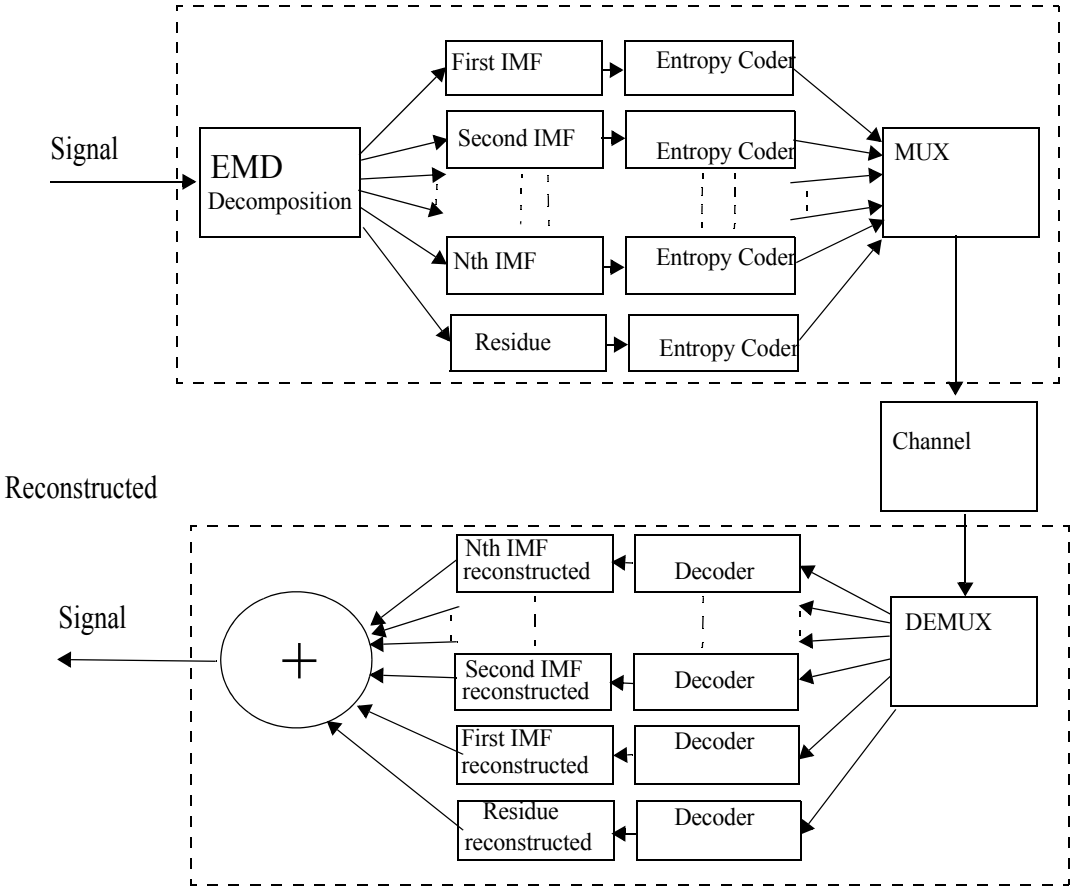


Figure 6.1. The EMD Entropy coding scheme.

6.1.1 Results

The set of IMFs and residue is compressed using fine quantization and Huffman coding results in a bitrate of 5.49 bpp at PSNR of 58.95 dB for the first IMF, 4.96bpp at PSNR of 58.97 dB for the second IMF, 5.47 bpp at PSNR of 59.04 dB for the third IMF, 6.79 bpp at PSNR of 58.89 dB for the fourth IMF, and 7.52 bpp at PSNR of 58.88 dB for the residue. Compression result using quantization levels ranging from 256 levels down to 2 levels are presented in Figure 6.2. An example of quantized IMFs and residue is shown in Figure 6.3 together with information on quantization, rate and distortion. We present the quantized IMFs and residue together with the original. The addition of the chosen reconstructions creates the reconstructed image in Figure 6.4 with 15.61 bpp and PSNR 33.9 dB.

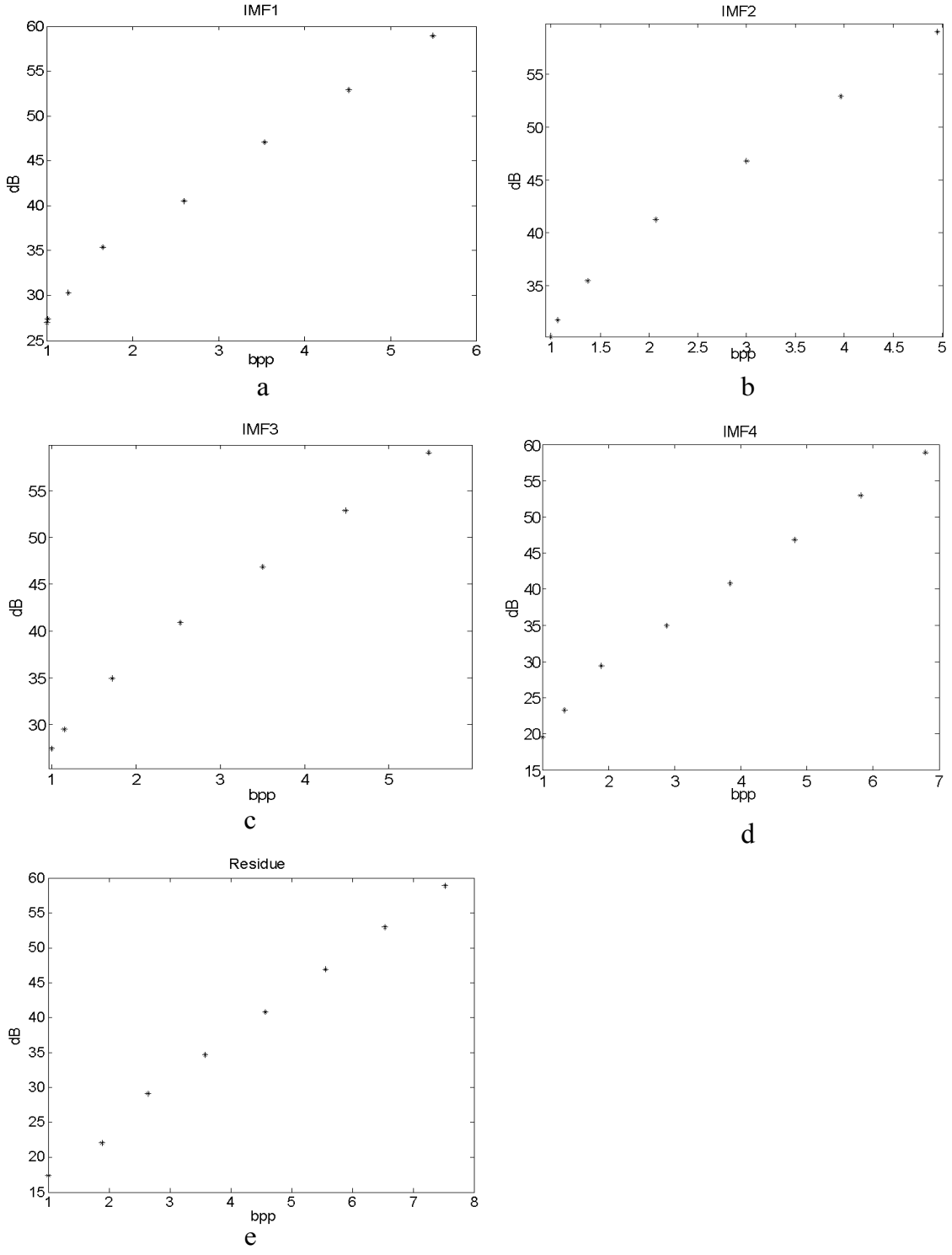
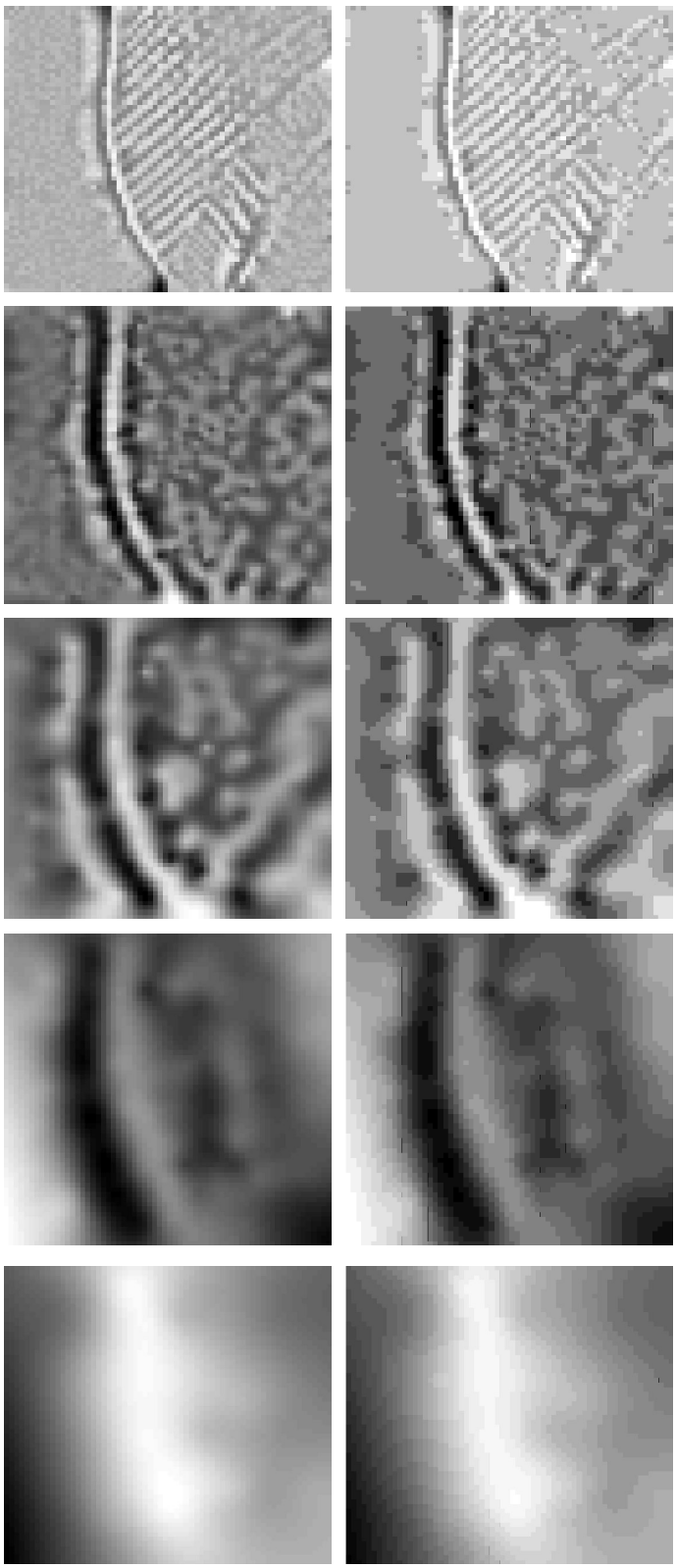


Figure 6.2. Entropy Coding results for each IMF and Residue, a) first IMF, b) second IMF, c) third IMF, d) fourth IMF, e) residue.



IMF1
40.54dB
2.60 bpp
32 quantization levels

IMF2
41.29 dB
2.07 bpp
32 quantization levels

IMF3
40.87 dB
2.53 bpp
32 quantization levels

IMF4
40.84 dB
3.84 bpp
32 quantization levels

Residue
40.80 dB
4.57 bpp
32 quantization levels

Figure 6.3. Left: original, the image on top and IMFs and the residue below. Middle: The quantized IMFs and residue. Right: Information on quantization levels, rate, and distortion for the images.

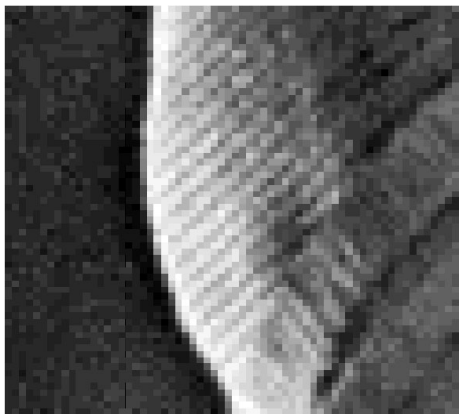


Figure 6.4. Reconstruction using 15.61 bpp results in PSNR 33.9 dB.

6.2 Extrema point coding

Our next idea to compress the EMD will be termed the extrema point coder.

The idea is to decompose the signal into its IMFs and transmit only the extrema points for each IMF. Then we reconstruct the IMF in the decoder with spline interpolation. This was first presented in [51]. The scheme is shown in Figure 6.5.

6.2.1 One-dimensional extrema point coding

To get a feeling for the behaviour of the IMFs in a coding application, we use the composite chirp test signal from example 5.1 in section 5.1. The IMFs are computed using cubic spline interpolation. The bitrate is estimated through entropy calculations by Eq. 3.1) in which the signal is quantized to the nearest integer. With the representation used here this means quantization to 256 levels. The entropy of a signal serves as an indicator of performance of a memory-less and lossless compression algorithm. The entropy is calculated for the IMFs themselves, their extrema vector, and their quantized extrema vector, as listed in Figure 6.6. The sum of entropy of the IMFs is much higher than the entropy of the signal itself. Yet, choosing to use only the extrema points gives the opportunity to code very sparse signals. The IMF extrema vectors are plotted in Figure 6.7. As can be seen, both in the plots and in the listing in Figure 6.6, the number of extrema points is low compared to the length of the original signal to code. The fewer extrema points to code the better. The cost of coding these sparse signals is estimated by the entropy of the extrema vector in Figure 6.6 and even more compression can be achieved with further quantization. in The example quantization to 128 levels is used. The entropy of the test signal is 7.08 bits/sample, while the sum of the entropies of the IMFs equals 28.52 bits/sample. Using the extrema points of the IMFs for representing the signal gives a summed entropy of 5.90 bits/sample, while using quantized extrema points leaves us with a summed entropy of 3.42 bits/sample.

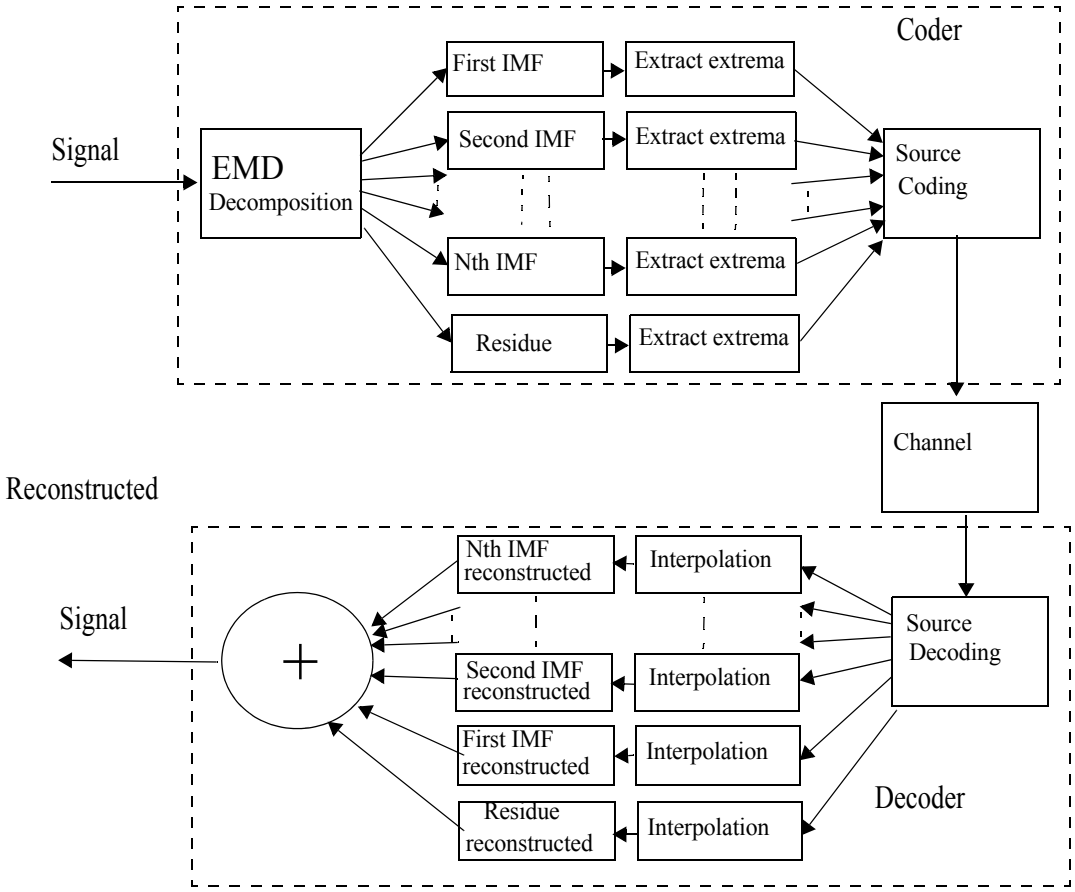


Figure 6.5. The extrema point coding scheme.

Signal	# sample	entropy	# extrema	entropy for extrema	entropy for quantized extrema
Composite chirp	2001	7.08			
IMF 1	2001	7.19	637	2.72	2.03
IMF 2	2001	6.73	345	1.56	0.99
IMF3	2001	5.13	189	0.68	0.23
IMF4	2001	3.37	101	0.46	0.11
IMF5	2001	2.69	54	0.28	0.04
IMF6	2001	3.41	33	0.20	0.02
SUM of IMF entropy		28.52		5.90	3.42

Figure 6.6. Listing of entropies of the IMFs and their extrema vectors.

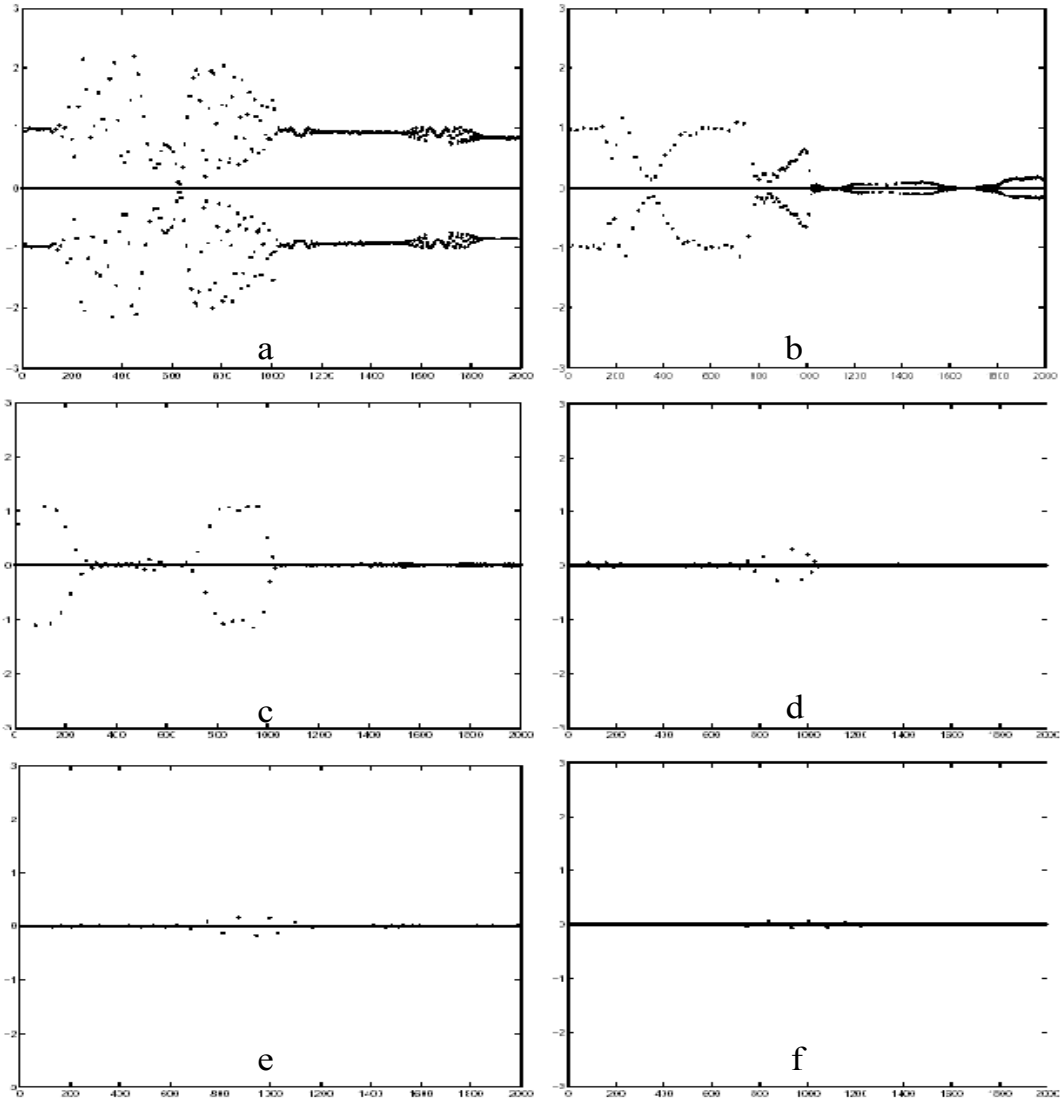


Figure 6.7. The extrema points of the IMFs respectively: a) IMF1, b) IMF2, c) IMF3, d) IMF4, e) IMF5 f) IMF6.

Reconstruction of the IMF is done by cubic spline interpolation of the extrema points received. The signal can then be reconstructed by the addition of the reconstructed IMFs and the residue. The interpolation introduces errors which are plotted in Figure 6.8. This is because the IMF itself is not a spline function but approximated by a spline interpolation of the extrema points when reconstructed. When no quantization is done on the extrema vectors, the cubic spline interpolation causes the least reconstruction error of several tested interpolation methods. This is in line with the examination of different interpolation methods in section 5.1.3.

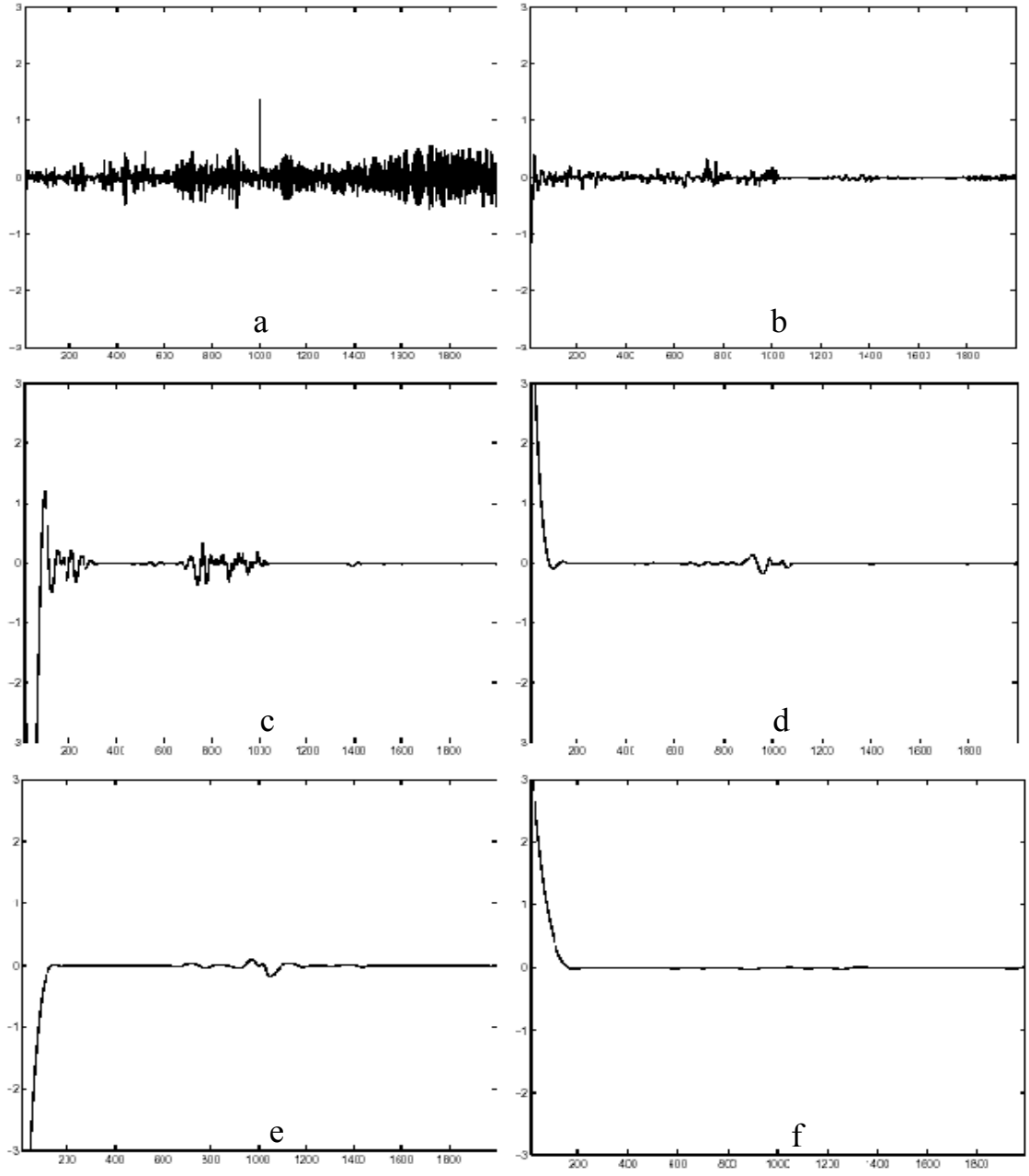


Figure 6.8. The reconstruction error for the IMFs respectively, a) IMF1, b) IMF2, c) IMF3, d) IMF4, e) IMF5 f) IMF6.

Comments

The lossless compression algorithm ZIP compresses the whole set of IMFs into approximately the same number of bits as when applied to the original signal. This indicates that there could be a way to overcome the redundancy in the set, and use this for compression. The spline interpolation of the extrema points introduces large errors in the reconstructed signal. This is particularly true for the first IMF which holds the most unsmooth part of the signal.

6.2.2 Two-dimensional extrema point coding

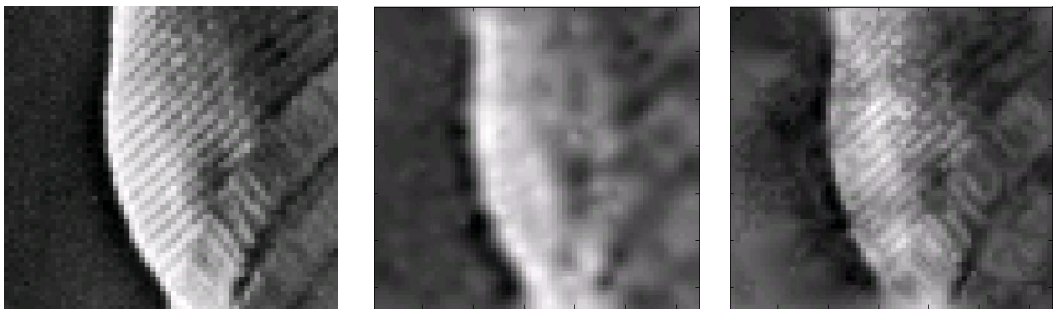
With the result of extrema point coding of time signals in mind, we want to use the same concept on images.

Coder outline

The two-dimensional extrema point coding is done by letting the extrema points of each IMF and residue represent the image. The coding follows the same scheme as for one-dimensional signals shown in Figure 6.5. The image is decomposed into a number of IMFs and a residue. For each IMF and the residue the extrema points are extracted and coded. Reconstruction is done with spline interpolation, either triangle-based cubic or thin-plate. The first IMF is not particularly smooth and the number of extrema points of the first IMF is very large. The smoothing thin-plate interpolation cannot reconstruct the first IMF in a proper manner. Instead we use the triangle-based cubic spline interpolation for the reconstruction of the first IMF.

The positions of the extrema points are coded using runlength coding and their values are quantized and fixlength coded. The exact positioning of the extrema point is extremely important for the result. In the extrema point coding scheme we use Elias coding for the runlength coding of the extrema point positions and fixlength coding for the quantized amplitudes of the extrema points. The fixed length is in relation to the number of quantization levels.

Example 6.1. Reconstructing the image with and without the cubic interpolating reconstruction of the first IMF gives very different visual performance as shown in Figure (a). Here the thin-plate spline interpolation is used for reconstruction of all IMFs and the residue. In Figure (b) the thin-plate spline interpolation is used for reconstruction of all IMFs and the residue, except for the first IMF, where the triangle-based cubic spline interpolation is used.



a b c

a) The test image “Detail one”, b) Reconstruction by thin plate interpolation of all IMFs and residue, PSNR 23.8dB. c), Reconstruction by thin plate interpolation of all IMFs except for the reconstruction of the first IMF where cubic spline is used, PSNR 21.0dB.

It is obvious that the visual quality is better in the latter case, although the PSNR measure tells us the opposite.

Example 6.2. The significant extrema points of the first IMF of the image in Figure (a), quantized with enlarged zero zone, are shown in Figure 6.9. The length histogram of zero runs in a line scan of the significant extrema point image is plotted in the histogram in the same figure.

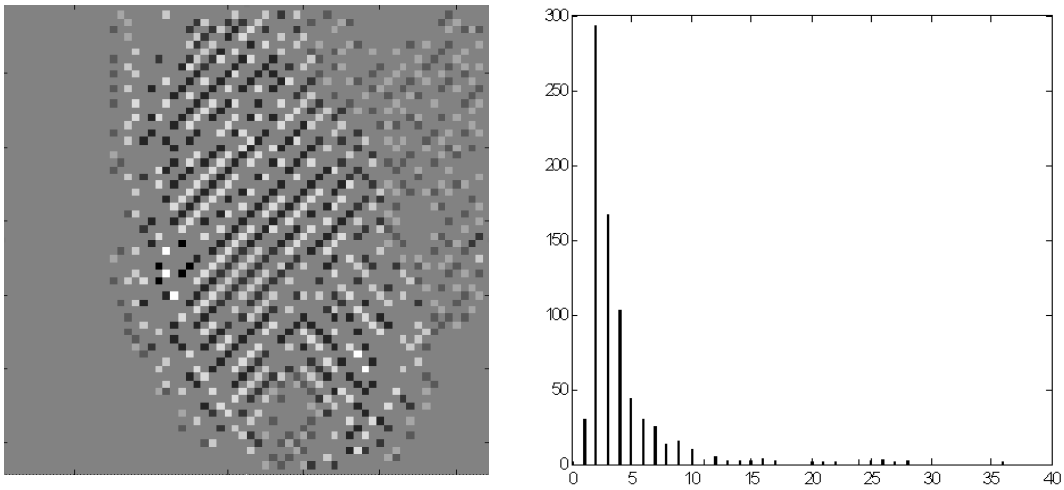


Figure 6.9. a) Extrema points of the first IMF, grey is zero. b) Length histogram of zero runs

Using Elias code gives us an effective way to assign codewords to the runlength values, giving 2982 bits to code the positions in this particular case. The fixed length code for coding of amplitude values uses here a 4-bit code to code 16 levels of amplitude. In this example the 767 nonzero positions can be coded with 767*4=3068 bits.

6.2.3 Results

The extrema points of the IMFs and the residue from the EMD of the test image in Figure 5.36 are shown in Figure 6.10.

The set of IMFs and residue is compressed using the extrema points shown in Figure 6.10. Five different sizes for the extended zero zone are used and the number of quantization levels ranges from 256 levels down to 2 levels.

The curves in Figure 6.11 show that each IMF and the residue can be represented with an acceptable distortion with a rate below one bit per pixel, except for the first IMF and maybe the second. In Figure 6.12 we have chosen to present one reconstruction of each IMF and residue. We present the reconstructed image together with the original. The addition of the chosen reconstructions creates the reconstructed image with 6.75 bpp at psnr equal to 19.79 dB.

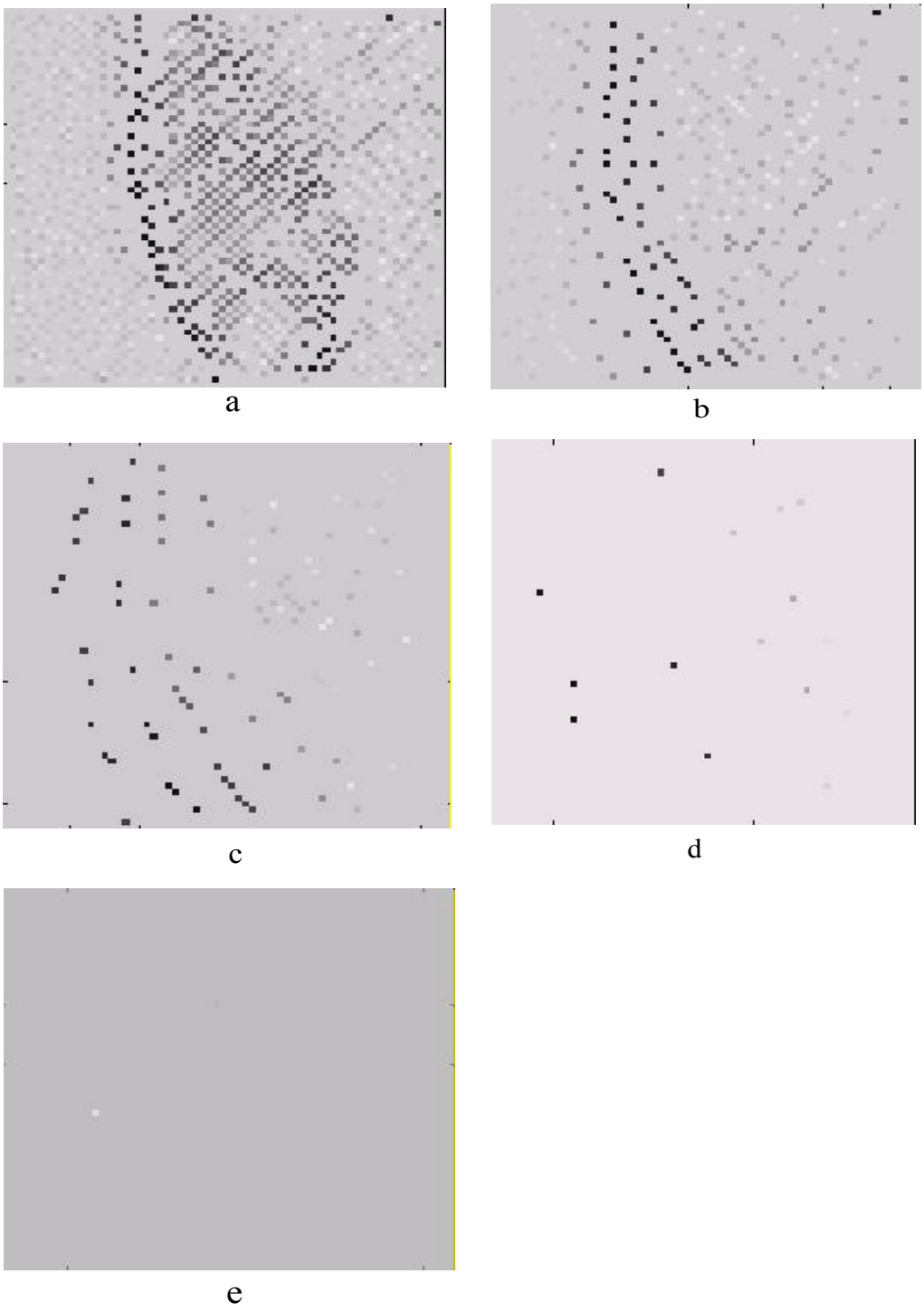


Figure 6.10. From the EMD of Figure 5.36: a) extrema points of IMF 1, b) extrema points of IMF 2, c) extrema points of IMF3, d) extrema points of IMF 4, extrema points of the residue.

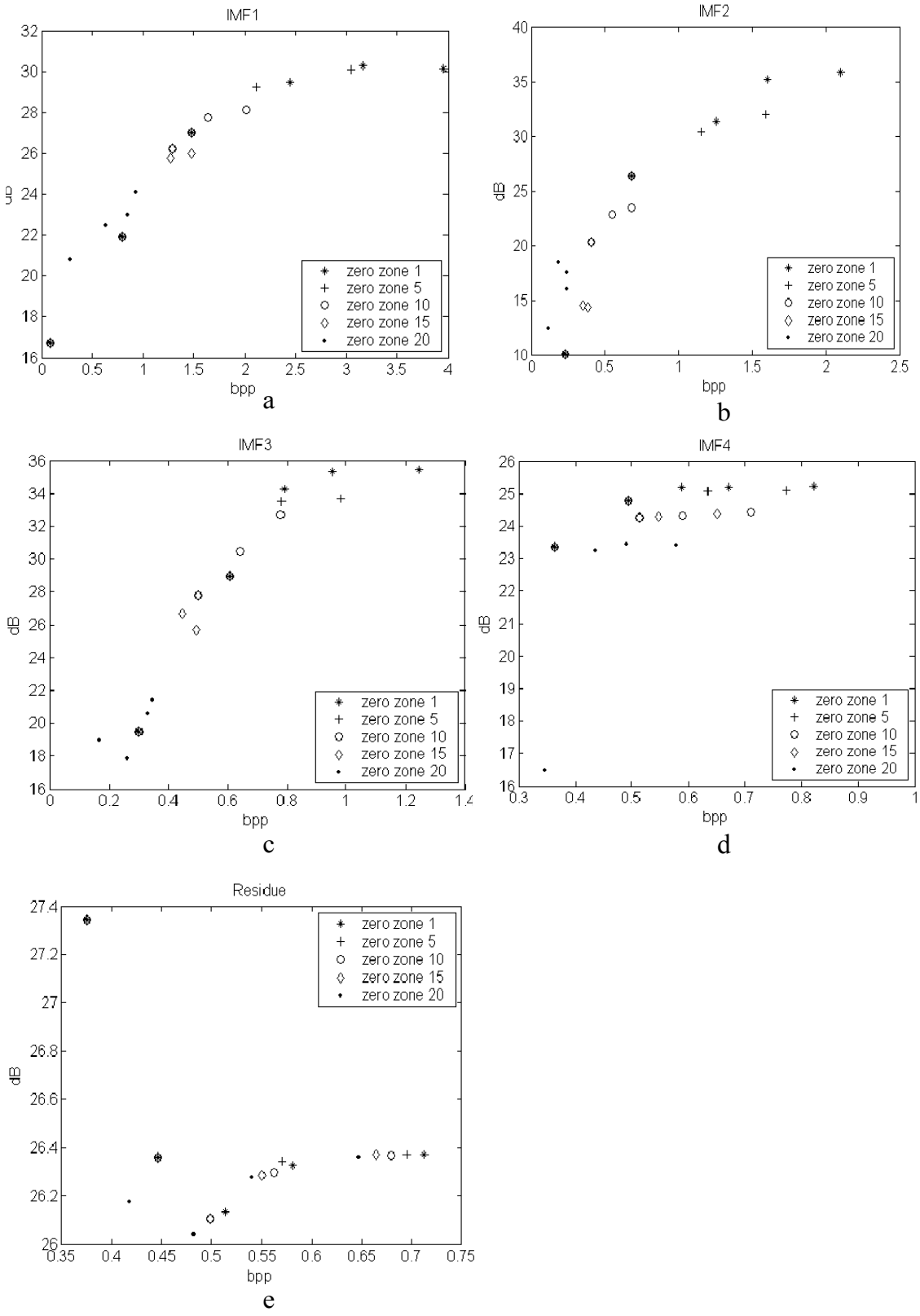
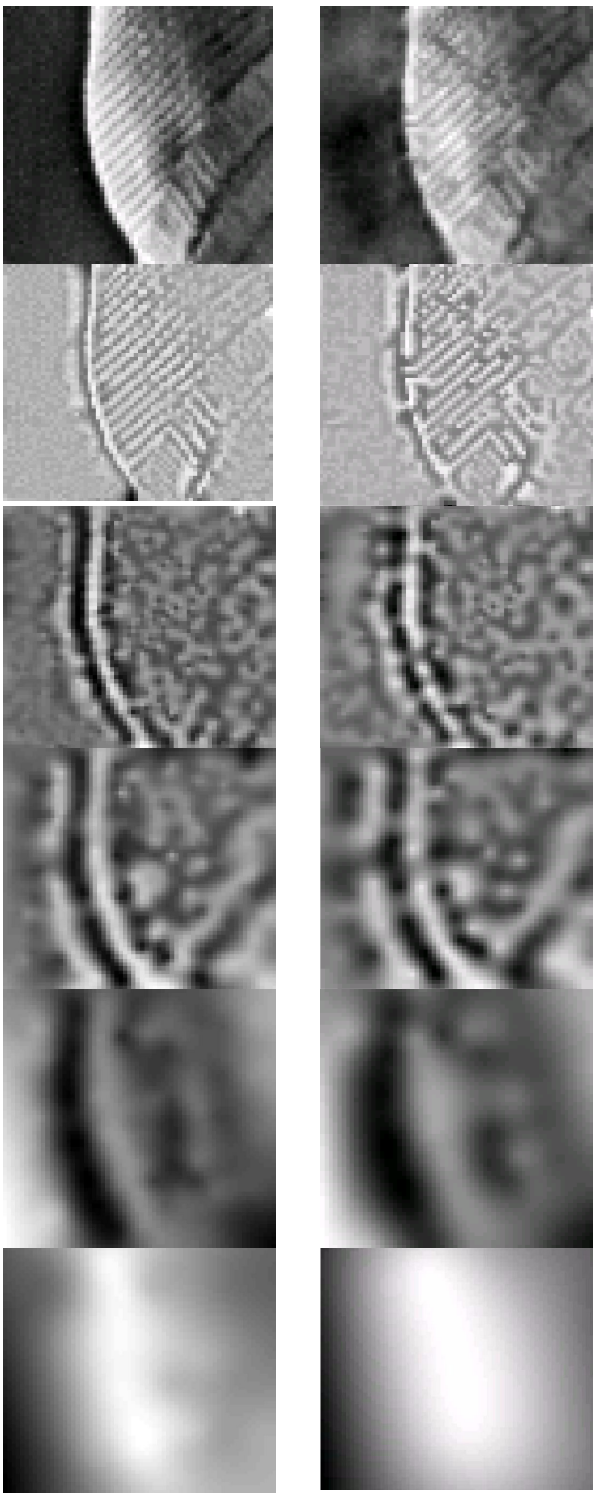


Figure 6.11. Extrema Point Coding results for each IMF and residue, a) first IMF, b) second IMF, c) third IMF, d) fourth IMF, e) residue.



Sum of IMFs
and residue
19.79 dB
6.75 bpp

IMF1
30.27dB
3.16 bpp
1227extrema points

IMF2
35.21 dB
1.60 bpp
461 extrema points

IMF3
35.35 dB
0.95 bpp
162 extrema points

IMF4
25.21 dB
0.67 bpp
38 extrema points

Residue
27.34 dB
0.375 bpp
6 extrema points

Figure 6.12. Left: original, the image on top and IMFs and the residue below. Middle: the reconstruction of the object to the left by interpolation of the extrema points. Right: information on rate and distortion for the reconstructed images. The number of extrema points includes extra helping points at the image border.

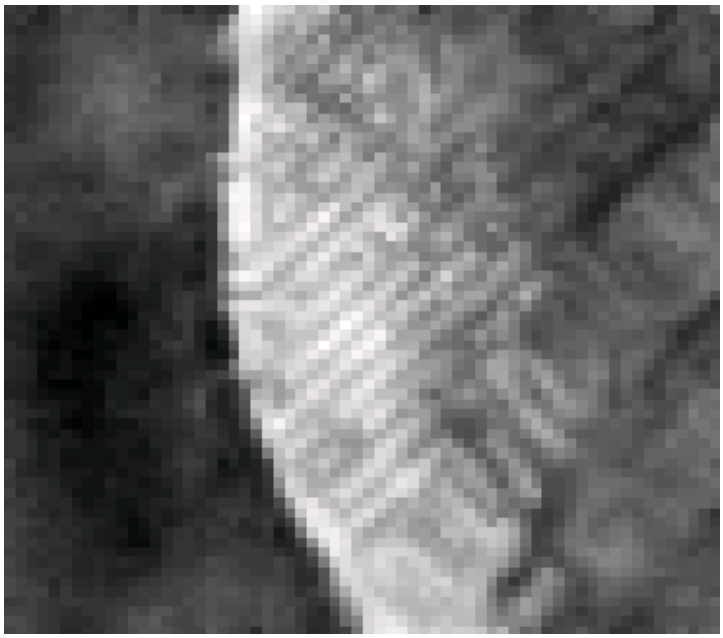


Figure 6.13. Reconstruction using 8 bpp results in PSNR 22.78 dB.

6.2.4 Conclusion

The extrema point coding approach is an attractive way to represent the image but does not seem to be suitable for compression. Normally in other coding methods it is in the quantization process that the major reduction of data and quality loss is found. In this method the major distortion comes from the reconstruction by interpolation. However, we have almost reached our first goal of representing the image without too much distortion at its original bitrate, Figure 6.13.

6.3 Block-based single-component DCT coding

We saw in Chapter 6.2 that using the extrema points to represent the image results in a large number of coefficients to code. This number has to be reduced before we can hope to achieve good compression. In this chapter we use the DCT, presented in section 1.2.1, for this purpose.

6.3.1 Coder outline

The sifting process for finding the IMFs sorts out one empiquency for each position in each IMF. This should trig a transform coder to produce a very compact code, as there is only one empiquency present in the IMF at each position. Most IMFs would only have response in the low frequency band of the DCT.

The idea is to divide each IMF into blocks of a specific size and use the DCT on each block. We use only one coefficient for each block. The motivation for this is the assumption that the IMF holds locally only one spatial frequency. If the DCT can represent this spatial frequency with only one coefficient, this would yield a good compression of the IMF. Decoding of the IMF is done with inverse DCT.

The coder is implemented in accordance with the scheme in Figure 6.14, which is based on the scheme in Figure 6.5.

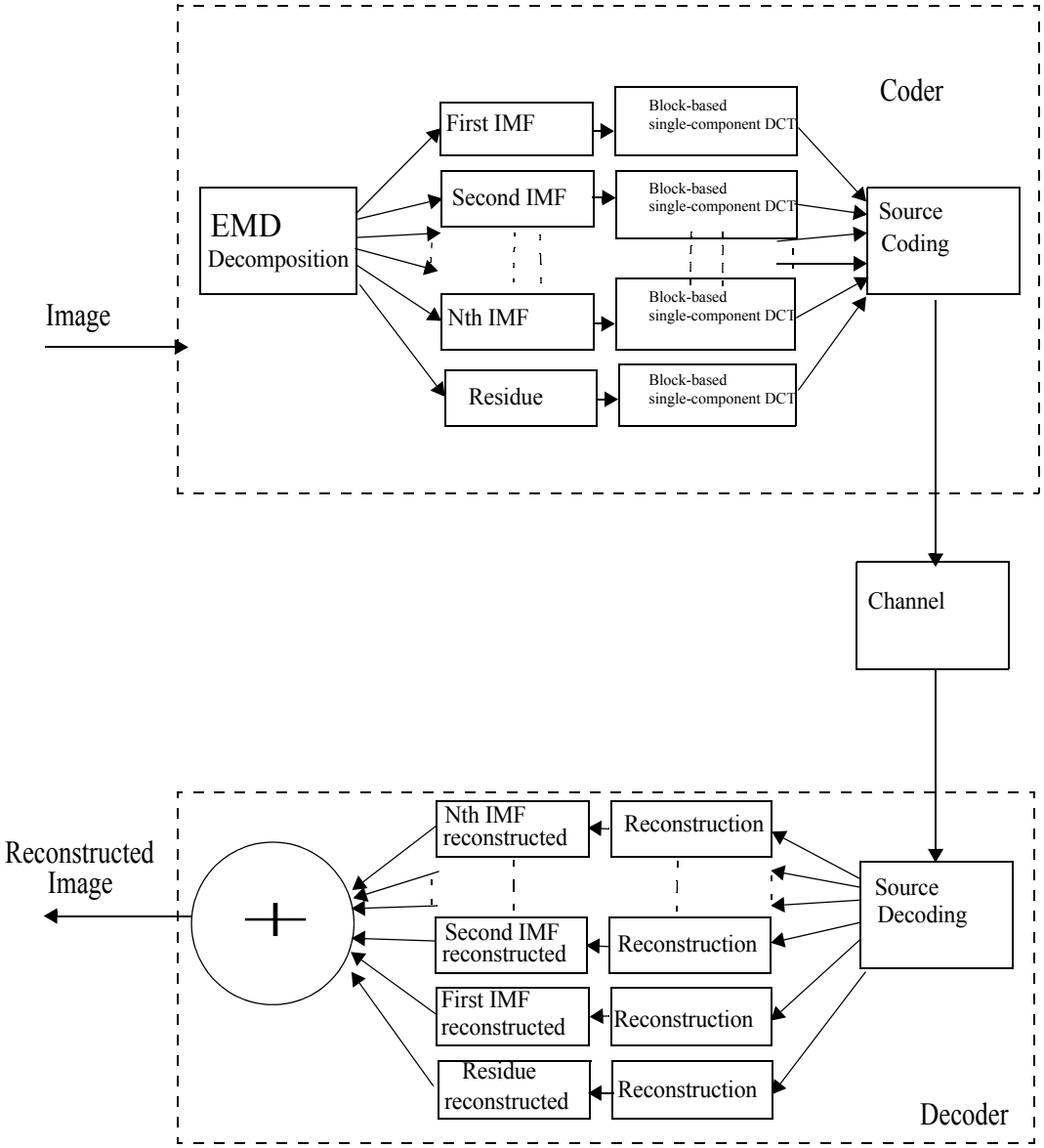


Figure 6.14. Single-component DCT coding scheme.

The implementation of the single-component DCT coding block is illustrated in Figure 6.15. First the IMF is divided into blocks of a specific size. Each block is transformed using DCT. The largest magnitude is detected and the rest of the coefficients are zeroed. The last step in this block is to patch the DCT blocks together into one large matrix, the coefficient image.

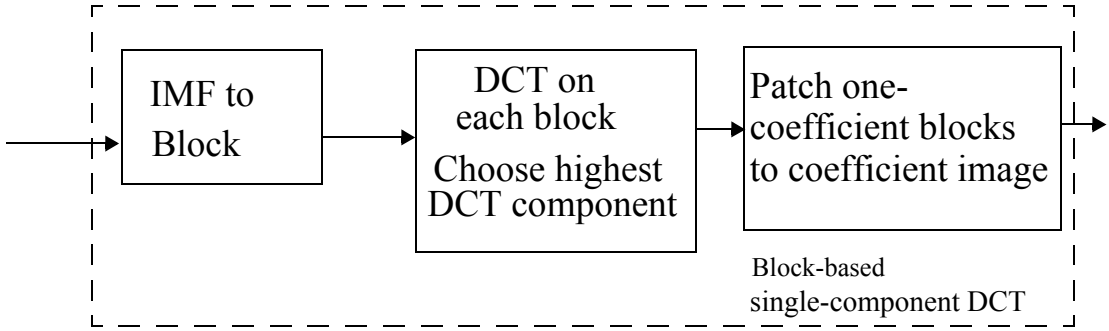


Figure 6.15. Detail of the block-based single-component DCT coding block.

Huffman code is used to represent the amplitudes of the DCT coefficient which are quantized to 256 levels. The positions are line scanned and runlength coded using Elias code.

The details of the reconstruction block are shown in Figure 6.16. The first step in the reconstruction is to divide the coefficient image into blocks holding only one non-zero value. Next we reconstruct the image block with inverse DCT, and patch the image blocks together to an image.

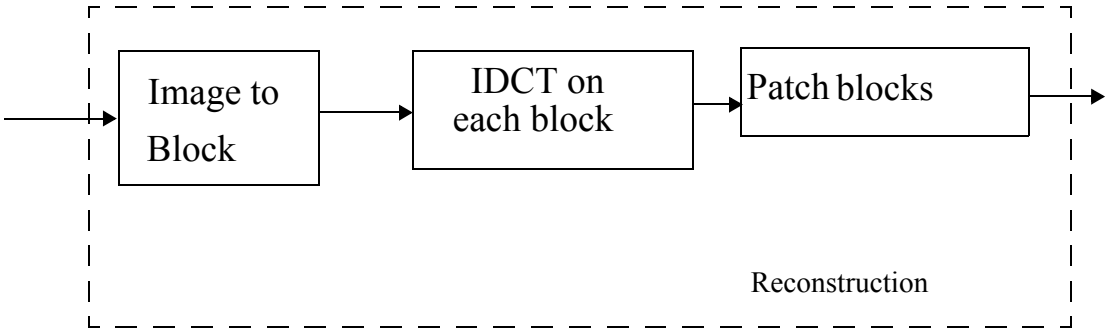
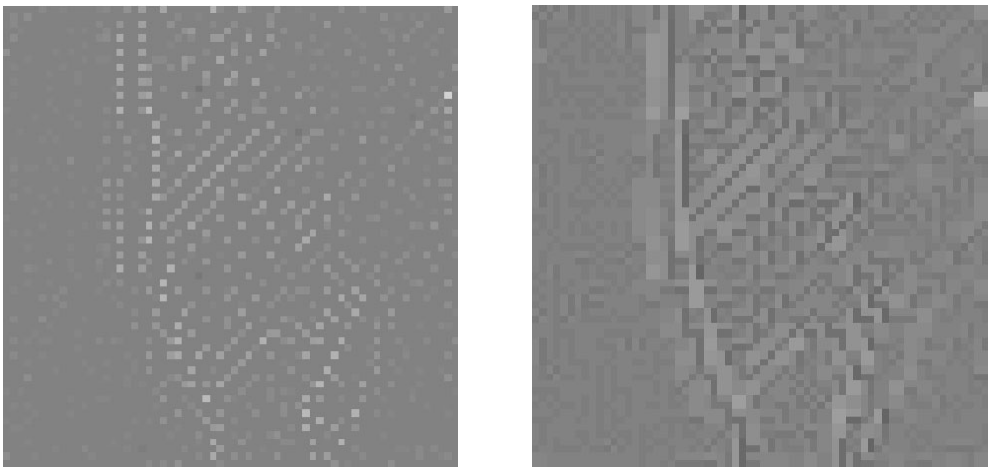


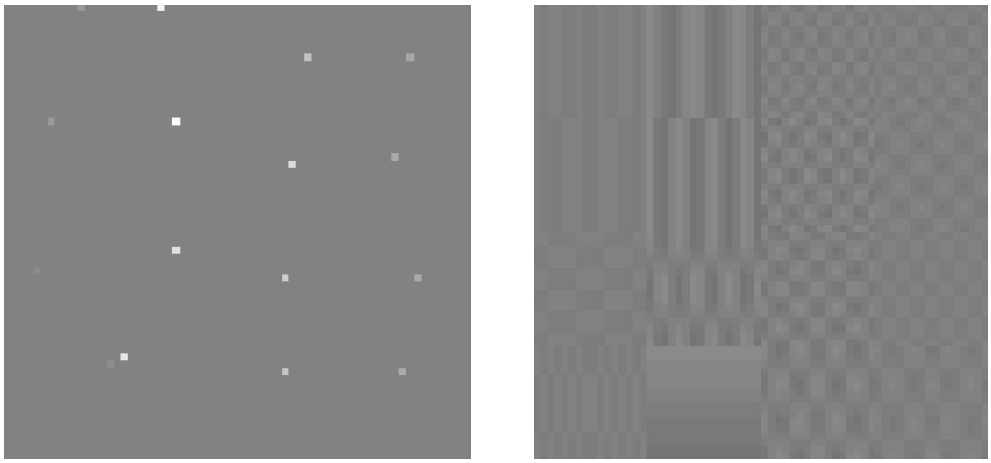
Figure 6.16. Detail of the reconstruction block.

Example 6.3. We show the effects of using two different block sizes for the first IMF in Figure 6.17 and Figure 6.18. In Figure 6.17 the first IMF is coded using blocksize 2x2. We see in (a) that the single component of each block has different positions within the block. This results in the trigging of different basis functions in the reconstruction in Figure 6.17(b). In Figure 6.18 the first IMF is coded using blocksize 16x16. The different positions are shown in Figure 6.18(a) and the reconstruction is shown in Figure 6.18(b). With this blocksize it is even clearer that different positioning of the single component gives different reconstruction results. The same basis function is used for the whole block, clearly seen in Figure 6.18(b). The high spatial frequency in the IMF is captured using both blocksizes, but in the first case the localization is more precise.



a b

Figure 6.17. First IMF, blocksize 2x2, a) DCT-components, b) reconstructed IMF,



a b

Figure 6.18. First IMF, blocksize 16x16, a) DCT-components, b) reconstructed IMF

6.3.2 Result

The different blocksizes tested on each IMF and the residue are 2x2, 4x4, 8x8, 16x16, and 32x32 pixels. In Figure 6.19 we can see the result of reconstructing the image by the addition of the reconstructed IMFs and residue when the single-component block-based DCT is used. Each dot represents a combination of blocksize from 2x2 to 32x32 for the four IMFs and the residue needed to reconstruct the image. Excluding the option for blocksize 2x2 eliminates the higher bitrates. Excluding the option for blocksize 32x32 as well eliminates the worst images in terms of PSNR. In this test we skipped coarse quantization of the amplitudes as this procedure only further reduces the performance in terms of PSNR. The method does not perform better than 20 dB due to blocking artifacts and the lack of precision in frequency response from the single-component DCT representation.

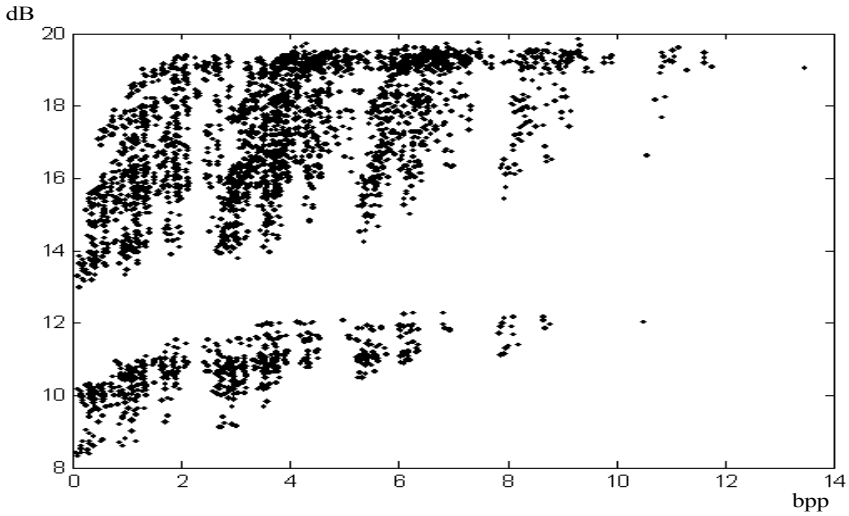
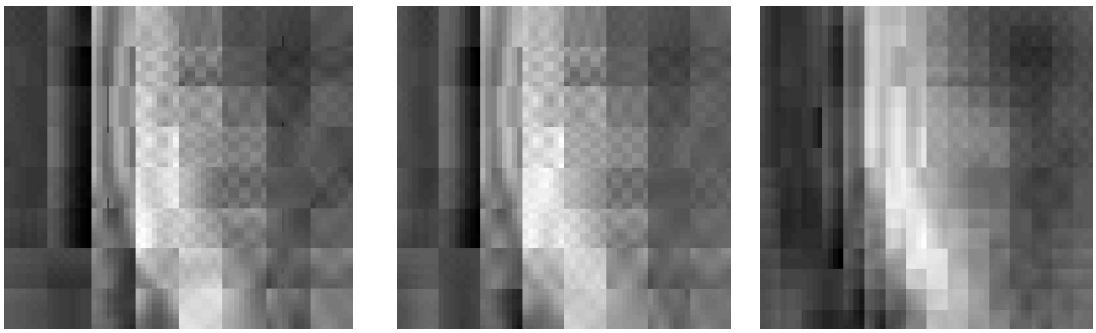


Figure 6.19. Each dot represents a combination of blocksizes from 2x2 to 32x32 for the four IMFs and the residue needed to reconstruct the image.

In Figure 6.20 some reconstructions are shown. The vector represents the blocksizes used for each IMF (first IMF, second IMF.....residue), where 5 is 32x32blocks, 4 is 16x16 blocks, 3 is 8x8 blocks, 2 is 4x4 blocks, 1 is 2x2 blocks. The best performance in terms of PSNR is shown in Figure 6.21, when we use many 2x2 blocks, but then we have higher bitrates instead.



a b c

Figure 6.20. a) (3 4 3 3 3) PSNR =17.7624 0.9810 bpp, b) (3 4 5 3 3) PSNR =17.5502 0.7686 bpp, c) (5 4 5 2 2) PSNR =19.0502 1.6602 bpp

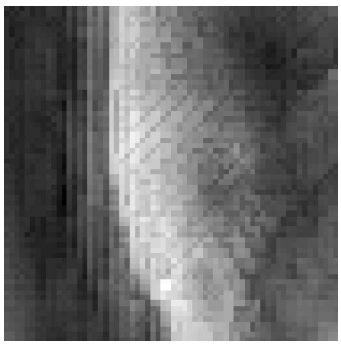


Figure 6.21. a) (1 3 2 1 1) PSNR =19.8284 9.3147 bpp.

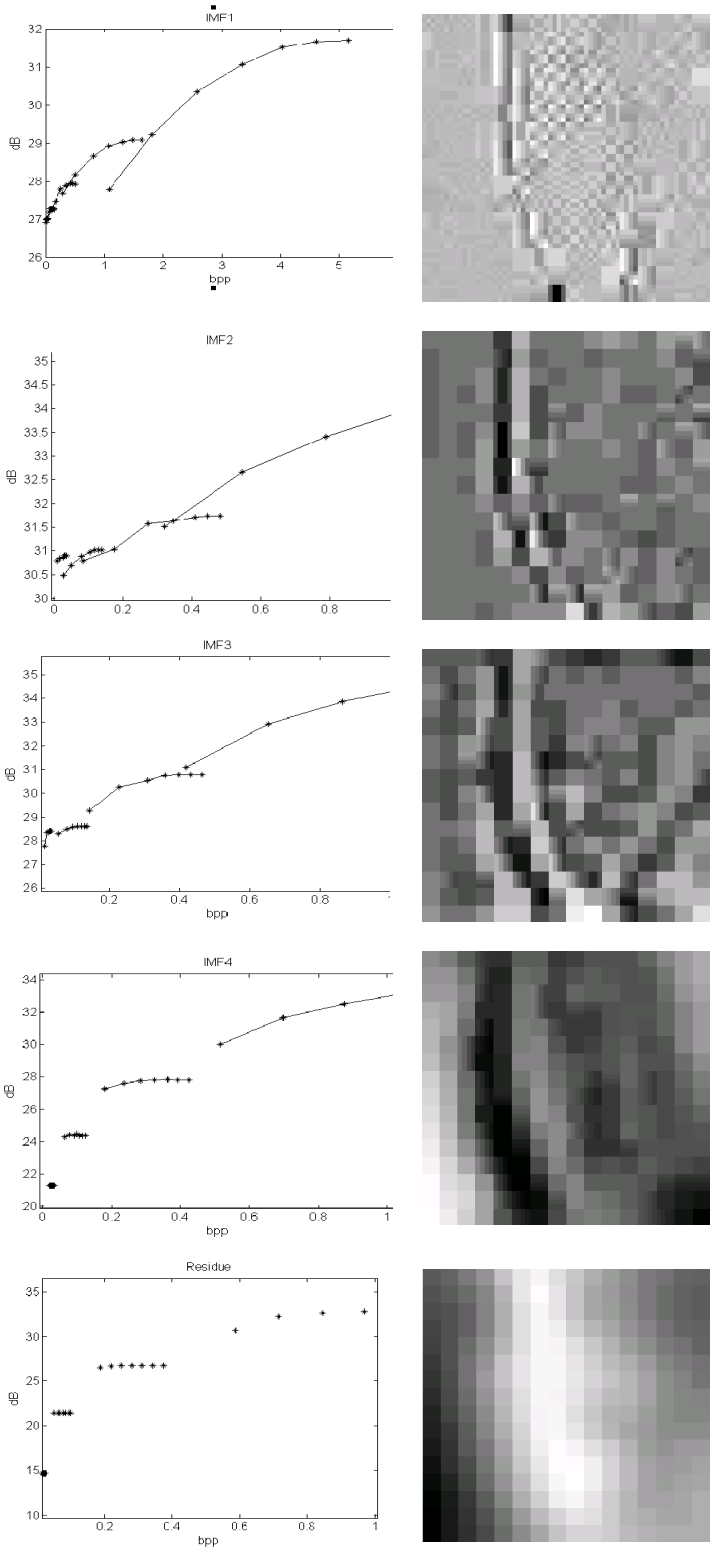


Figure 6.22. Result to left, in the middle a reconstructed example using the parameters specified to the right.

6.3.3 Two-coefficient block-based DCT

Even though the IMF could be locally zero mean, the blocking tiles it into non-zero-mean parts. We need to include the DC component of each block to get a representation good enough for a compression algorithm. Our method is the same as the one-coefficient block-based DCT but with the DC component added to get better performance. In this scheme the DCT components kept are also quantized using different numbers of quantization levels.

Result

Figure 6.22 shows compression results from the two-component block-based DCT. For bitrates below 1 bpp, the distortion is acceptable in terms of PSNR value for most of the IMFs except for the first IMF. But the visual performance shows severe blocking artifacts.

6.3.4 Conclusion

Due to the blocking artifacts, both the one-component block-based DCT and the two-component block-based DCT fail to be serious candidates for a compression algorithm. However, they illustrate several properties of the IMFs.

6.4 DCT threshold coding

The sifting process for finding the IMFs sorts out higher empiquencies in the first IMF. The tendency is that the IMFs other than the first are low frequency images. This can be seen in the DCT of the whole IMFs in Figure 6.23, in which the image's lowest frequencies in both directions are represented by the components in the upper left corner of the image and the image's highest frequencies in both directions are represented by the components in the lower right corner of the image. The upper right corner represents the high frequencies in the horizontal direction and low frequencies in the vertical direction, while the lower left corner represents the high frequencies in the vertical direction and low frequencies in the horizontal direction. In Figure 6.23(a) the components are spread all over the matrix, with a highlight in the middle representing the dominating frequencies of the first IMF. In Figure 6.23(b) the largest components are located along the upper edge, meaning that the second IMF is mostly low frequency in the vertical direction and has little higher frequency content in the horizontal direction. For the rest of the IMFs and the residue, the frequency components concentrate in the upper left corner, where they represent low spatial frequency.

6.4.1 Coder outline

The idea is to keep only the significant components located in the upper left corner. The size of the triangle of components is determined by a sliding limit line parallel to the di-

agonal from the lower left corner to the upper right corner. The limit line shall be as close to the upper left corner as possible when still no components larger than the threshold are to the right of the line. All the components in the upper left corner to the left of the line are undisturbed, only uniformly quantized.

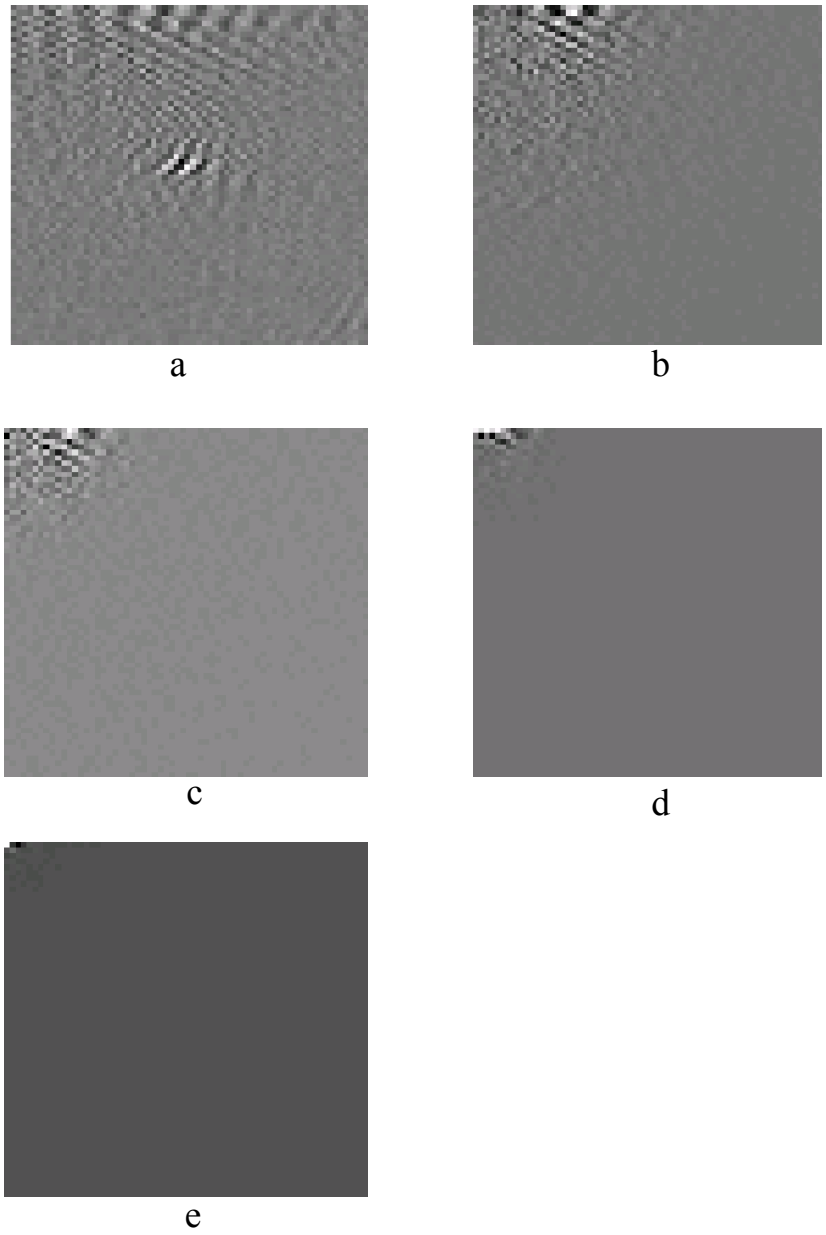


Figure 6.23. a) 64x64 DCT transforms of the 64x64-sized IMFs (a-d) and the residue (e) of the EMD shown in the left column in Figure 6.29.

6.4.2 Implementation

The implementation scheme is shown in Figure 6.24. The image is decomposed into its IMFs and the residue. Each IMF and the residue is fed into the coding block. After reconstruction the IMFs are added together with the residue to reconstruct the image. The details of the coding block and the reconstruction block are further described below.

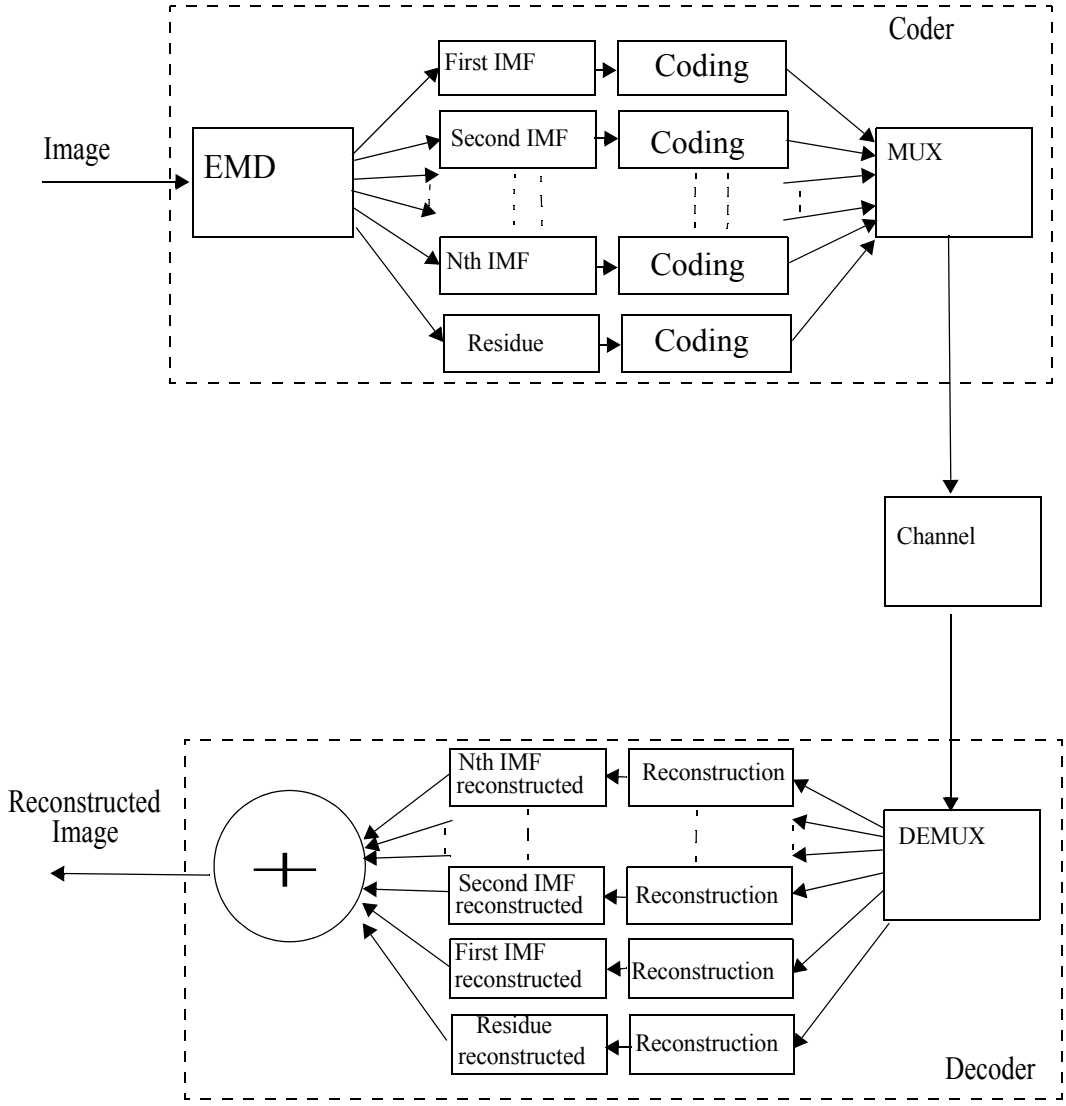


Figure 6.24. The DCT Threshold coding scheme.

The coding block

The details of the coding block are shown in Figure 6.26. Each IMF and the residue are transformed by the DCT. The resulting DCT component matrix is searched for the size of the upper left corner by scanning the component matrix in a zigzag pattern starting with the DC level coefficient in the upper left corner and stopping at the lower right corner. The scan pattern is illustrated in Figure 6.25. The scan is truncated at the position of the first component larger than the threshold found when scanned from the end. The resulting DCT components in the scan are uniformly quantized. The component values in the truncated scan are Huffman coded.

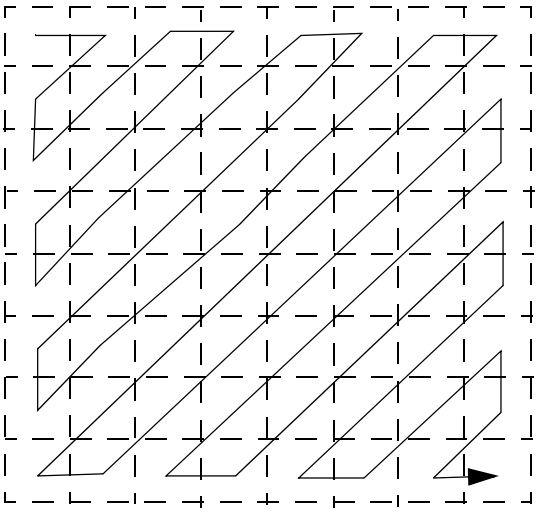


Figure 6.25. Zigzag scan pattern.

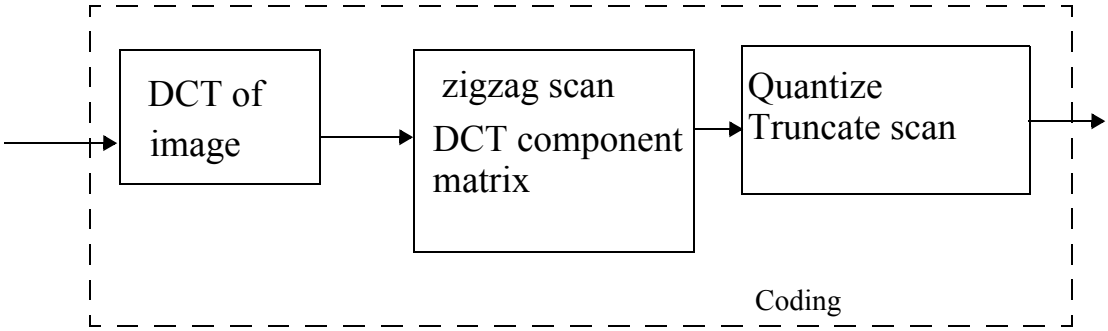


Figure 6.26. Detail of the coding block.

The reconstruction block

The details of the reconstruction block are shown in Figure 6.27. When the truncated and quantized scan is reconstructed from the source decoding step, the original length of the scan is established by adding zeros at the end. The next step is to resize the data into a matrix format. Finally the IMF is reconstructed by the inverse DCT.

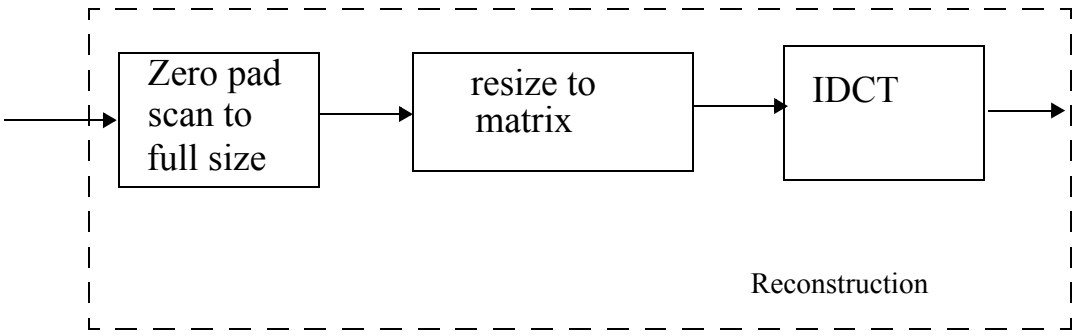


Figure 6.27. Detail of the reconstruction block.

6.4.3 Result

DCT Threshold Coding results for each IMF and residue are shown in Figure 6.28. The full result is presented to the left and to the right is a blow up of the result below one bit per pixel. For the first IMF no blow up is presented. Each graph consists of seven different curves, representing the seven different quantizations used in the test. The stars on each curve present the result for the different truncation lengths. In this test we use a number of predefined truncation lengths instead of a threshold-controlled truncation.

In Figure 6.29, we present one reconstruction of each IMF and the residue, together with the original and the parameters. The reconstructed image is in Figure 6.30 (a) at 28.26 dB and the sum of bitrate of. 2.26 bpp. In Figure 6.30 (b) the compression ratio is set to 1. The image is reconstructed at 34.41 dB using a sum of bitrate of 7.97 bpp where most of the bits, 4.2 bpp, 2.8 bpp, are spent on IMF1 and IMF2 respectively.

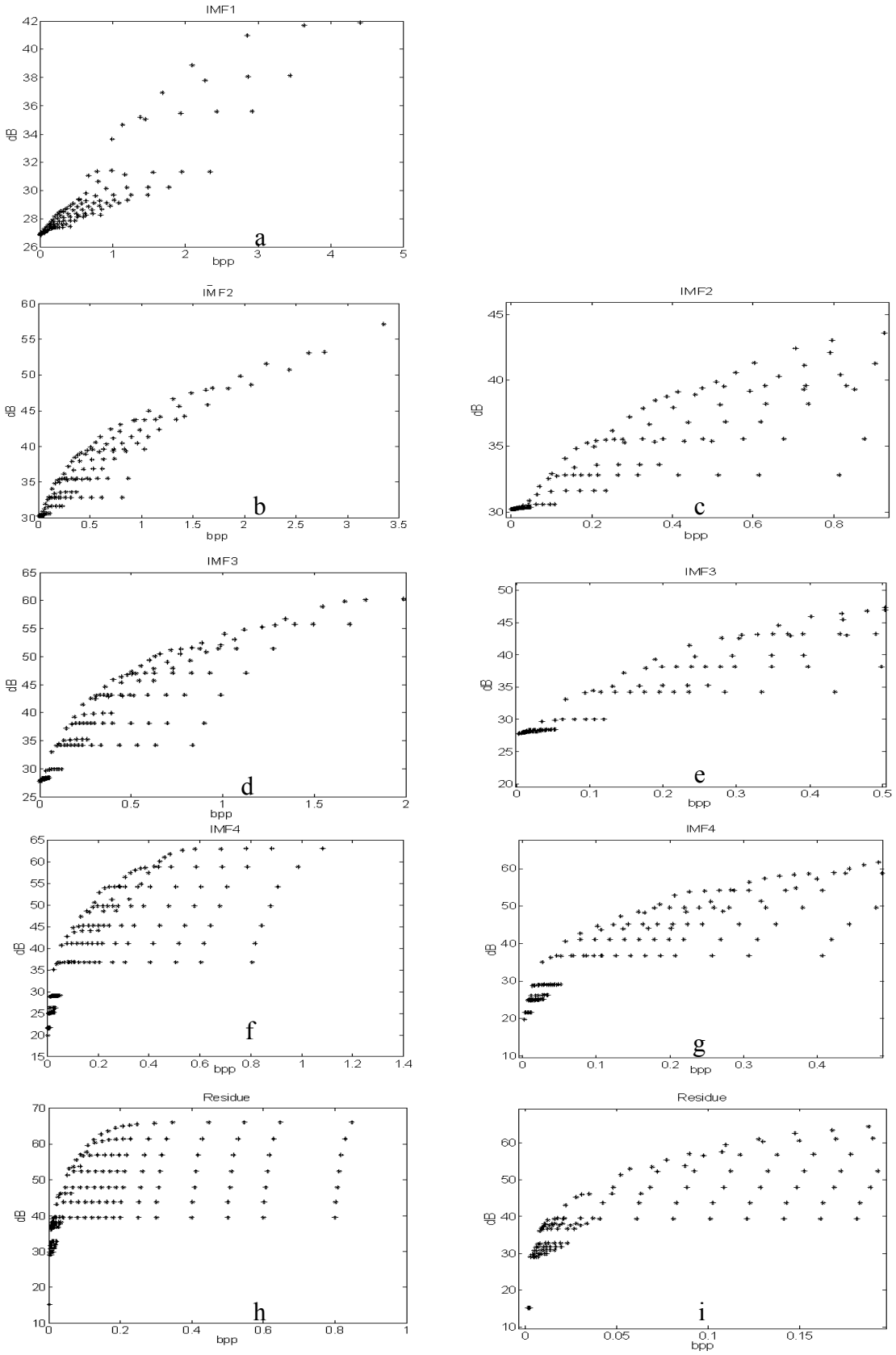
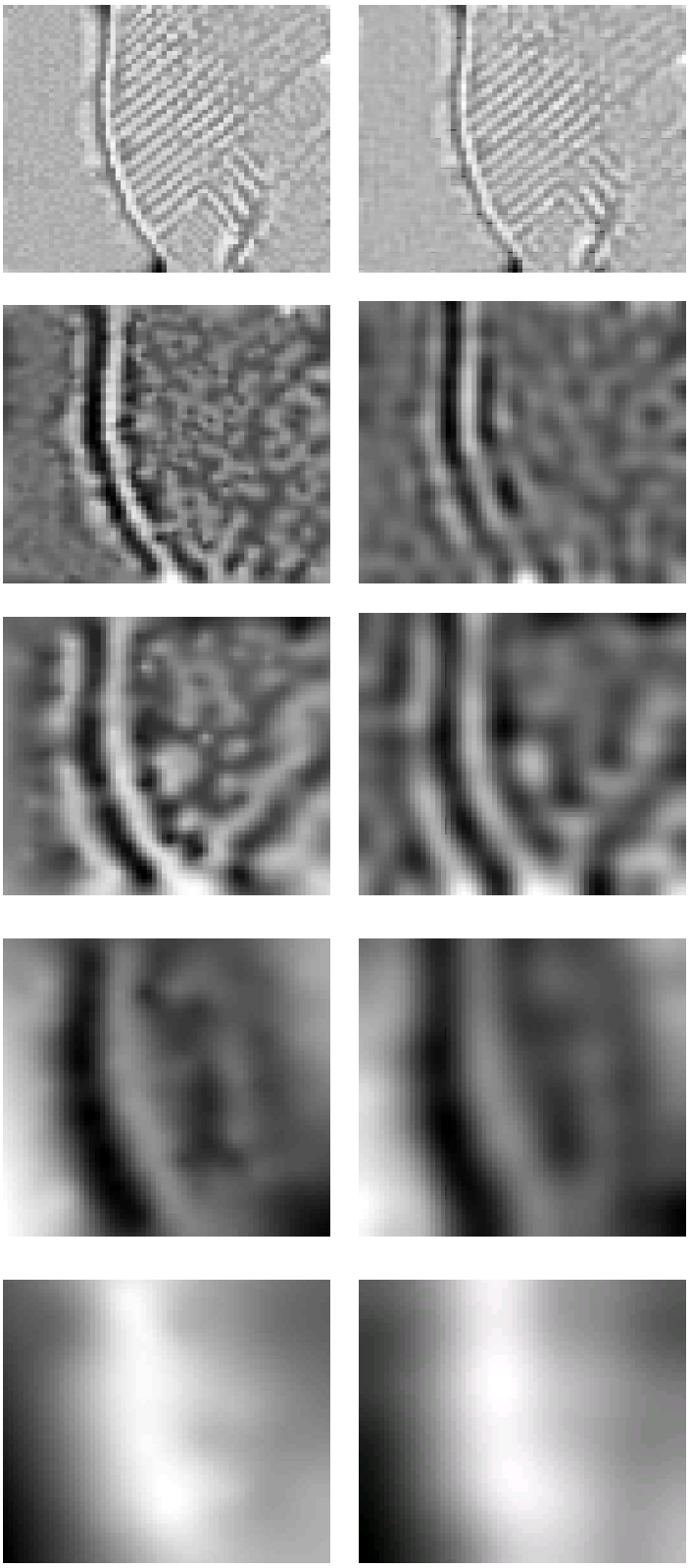


Figure 6.28. DCT threshold coding results for each IMF and residue, a) first IMF, b) second IMF, c) zoomed, d) third IMF, e) zoomed, f) fourth IMF, g) zoomed, h) residue, i) zoomed.



IMF1
35.47dB
1.94 bpp
length of vector:2048
64 quantization levels

IMF2
35.28 dB
0.28 bpp
length of vector: 328
32 quantization levels

IMF3
35.11 dB
0.13 bpp
length of vector: 164
16 quantization levels

IMF4
35.09 dB
0.027 bpp
length of vector:82
4 quantization levels

Residue
36.13 dB
0.008 bpp
length of vector: 29
4 quantization levels

Figure 6.29. Original to left, reconstructed in the middle, using the parameters specified to the right.

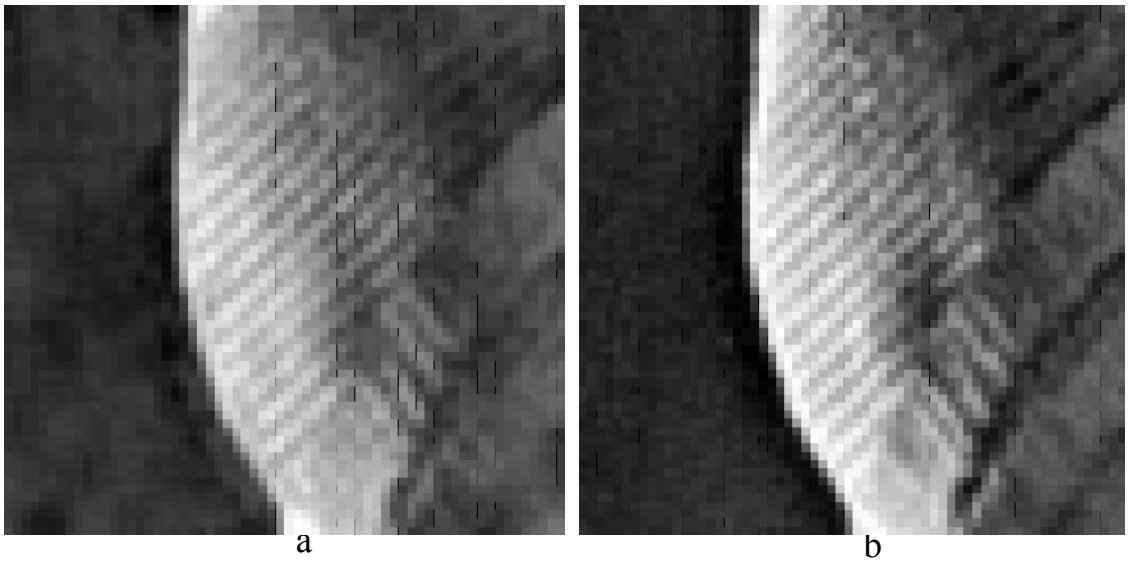


Figure 6.30. a) Reconstructed image, PSNR =28.46 dB, sum of bitrate = 2.39 bpp, b) Reconstructed image, PSNR =40.51 dB, sum of bitrate = 8.00 bpp.

When implementing a more traditional threshold coding of the DCT of the EMD, keeping only the significant components, the result is results for each IMF and residue are approximately the same as for the method presented here.

6.4.4 Conclusion

Threshold coding of the DCT of the IMF and residue gives a compact representation of these images except for the first and perhaps the second IMF which would need better methods to be efficiently compressed. Although the two coding methods tested result in different components kept, the coding result is not very different. The traditional threshold coder is slightly better. We have reached our first goal of representing the image with the EMD, using no more bits than the original, and without too much distortion. Using a more efficient way to code the first IMF we expect to achieve better image compression.

6.5 Summary

In this chapter different methods to compress the EMD are tested. First the set of IMFs and residue is entropy coded with both fine quantization and coarser.

The next idea to compress the EMD is the extrema point coder. The idea is to decompose the signal into its IMFs and transmit only maxima and mean values for each IMF. Then we reconstruct the IMF in the decoder with spline interpolation. The method is evaluated both for time signals and for images. The method is good for compression of the smooth IMFs, but fails to compress the first IMF due to its high-empiquency content. The extrema point coding approach is an attractive way to represent the image but is ill suited for compression. It is in the quantization process that the major reduction of data and quality loss normally appears. In this method the major distortion comes from the recon-

struction by interpolation. Using the extrema points to represent the image results in a large number of coefficients to code. This number has to be reduced to have a good compression algorithm.

Another idea is to divide each IMF into blocks of a specific size and use the DCT on each block. In our first attempt we use only one coefficient for each block. The motivation is the assumption that the IMF holds locally only one spatial frequency. If the DCT can represent this frequency with only one coefficient this would yield good compression of the IMF. Even though the IMF should be locally zero mean, the blocking tiles the IMF into non-zero mean parts. We need to include the DC component of each block to get a representation good enough for a compression algorithm. Due to the blocking artifacts, both the one-coefficient block-based DCT and the two-coefficient block-based DCT fail to be candidates for a compression algorithm. The two-component blockbased DCT representation gives us good results in terms of PSNR values and bitrates, but the blocking artifacts are annoying.

The tendency is that the IMFs other than the first are low frequency images. This can be seen in the DCT of the whole image. The idea in our third and fourth scheme is to keep only the significant components. This is done by truncation of a zigzag scan of the DCT matrix or only thresholding the components. Threshold coding gives a compact representation of the IMF and residue except for the first and maybe the second IMF. With this method we can represent the image with EMD, using no more bits than the original, and without too much distortion.

Chapter 7

Variable Sampling of EMD

In Chapter 6 our first goal was met even without finding a good method for the compression of the first IMF. To meet our second goal we must find a method for the first IMF. With the variable sampling presented in this chapter we go back to the thinking that lead us to the one-coefficient block-based DCT in section 6.3. A method of using overlapping 7x7 blocks is introduced to overcome blocking artifacts and to further reduce the number of parameters required to represent the image. The very special properties of the IMF that these are locally zero mean and ideally do not have more than one extrema point between two neighbouring zero-crossings enables us to use the empiquency to control the sampling rate of each block.

7.1 Variable sampling of overlapping blocks

The special property of the IMF that the empiquency varies can be used for variable sampling. Areas with many extrema points have high empiquency, while areas with a few or no extrema points have low empiquency. The IMFs are smoother than the image itself; only the first IMF holds the nonsmooth parts of the image. This means that it should be possible to subsample the IMFs. Due to the different empiquencies in the different parts of the IMF, the subsampling can be different in different parts of the IMF.

It is known [59] that a band-limited signal can be uniquely determined from its nonuniform samples, provided that the average sampling rate exceeds the Nyquist rate. The extrema points define the maximum empiquency in the IMF. Maximum empiquency is found by examining the space between the extrema points. In the first IMF there are areas where two neighbouring pixels both are extrema points, thus the maximum empiquency is 0.5. Letting this define our Nyquist rate, we expect that it is not possible to subsample

this IMF without distortion.

Our suggestion is to treat the IMF blockwise. This way the sampling rate for each block can be defined according to its empiquency content. The high empiquency blocks that cannot be subsampled according to our Nyquist rate are not modified. The remaining ones are subsampled.

7.1.1 Sampling of overlapping blocks

In the implementation of the blocking process we choose to use overlapping blocks of size 7x7 pixels. This is to minimize the artifacts from the blocking and to further reduce the samples used to represent the IMF. The corners of the block are always represented, regardless of the chosen sampling rate.

The overlapping pixels in two neighbouring blocks will be the same, but used twice, as shown in Figure 7.1. This will ensure that the concatenated blocks have the same values at the edge pixels. The overlapping pixels will only belong to one of the blocks when patching and counting the total number of samples.

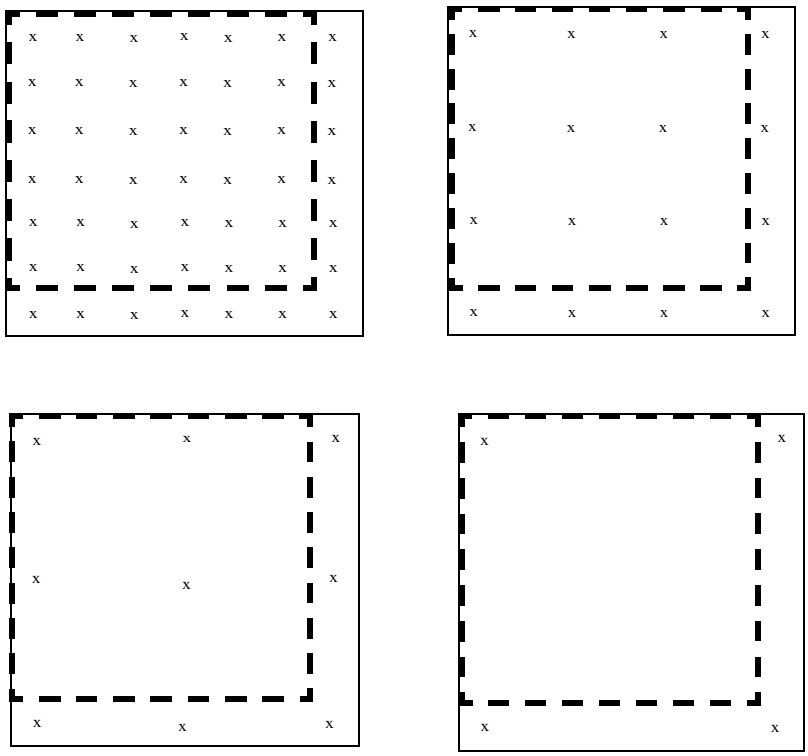


Figure 7.1. Variable sampling with overlapping 7x7 blocks.

The sampling pattern within a block consists of every pixel, every second pixel, every third pixel, and every sixth pixel in both directions, to represent 1/1, 1/4, 1/9, and 1/36 of the pixels, respectively, as shown in Figure 7.1.

To find maximum empiquency we use a separable approach, analysing the extrema points by row and column separately and choosing the maximum empiquency.

7.1.2 Interpolation

For a signal $f(t)$ that is band limited to $\left[-\frac{1}{2T},\frac{1}{2T}\right]$ and sampled according to the sampling theorem, the signal can be reconstructed without distortion from its sampled version $f(nT)$ through the reconstruction formula

$$f(t) = \sum_{n=-\infty}^{\infty} f(nT)h_T(t-nT) \tag{7.1}$$

A sampling function $h_T(t)$ used for reconstruction satisfies

$$h_T(t) = \begin{cases} 1 & if \quad t = 0 \\ 0 & if \quad t = nT \end{cases} \tag{7.2}$$

The sampling function that should be used according to the sampling theorem is the basis function *sinc(t)*.

$$h_T(t) = \frac{\sin(\pi t/T)}{(\pi t/T)} \tag{7.3}$$

This function has infinite duration.

The sampling theory relies on the use of ideal filters. The use of finite signals is in contradiction to the band limitation approach. An alternative approach is to use splines. These are piecewise polynomials with pieces that are smoothly connected together at the sample points.

The cubic B-spline interpolator is given by

$$b^{(3)} = \begin{cases} \frac{2}{3} - \left|\frac{t}{T}\right|^2 + \frac{\left|\frac{t}{T}\right|^3}{2} & 0 \le \left|\frac{t}{T}\right| < 1 \\ \left(2 - \left|\frac{t}{T}\right|\right)^3 & 1 \le \left|\frac{t}{T}\right| < 2 \\ 0 & 2 \le \left|\frac{t}{T}\right| \end{cases} \tag{7.4}$$

Considering the fact that we are treating finite signals, the use of the spline interpolator to reconstruct the image from its samples is superior to using a windowed sinc func-

tion as interpolator [86]. The uniformly sampled points of the block are thus connected by a surface created by the use of an interpolating cubic spline extended to two dimensions.

7.2 Implementation

The method of variable sampling of overlapping blocks is implemented according to the scheme in Figure 7.2. Each IMF and the residue from the EMD of the image are processed in the variable sampling block.

7.2.1 The variable sampling block

The details of the variable sampling block are shown in Figure 7.3. The extrema point image holds only the significant extrema points with respect to some size of the zero zone. Both the image itself and the extrema point image are decomposed into overlapping subblocks. The blocks of the extrema point image are used for the determination of the maximum empiquency in the block. This steers the sampling rate of the corresponding image block. In the resulting bitstream each block is marked with a two-bit header that holds information about the sampling rate.

7.2.2 The reconstruction block

The details of the reconstruction block are shown in Figure 7.4. The two-bit header of the stream tells us not only the sampling rate but also the length of the stream for the block. With this information the coefficients can be placed in their proper place in the 7x7pixel block. The stream only provides samples for the 6x6 block; the missing samples are found in the neighbouring blocks.

Reconstruction of the IMF uses the 7x7 block for block reconstruction with interpolation. The nonoverlapping 6x6 part of the reconstructed block is used when patching the blocks together to reconstruct the image.

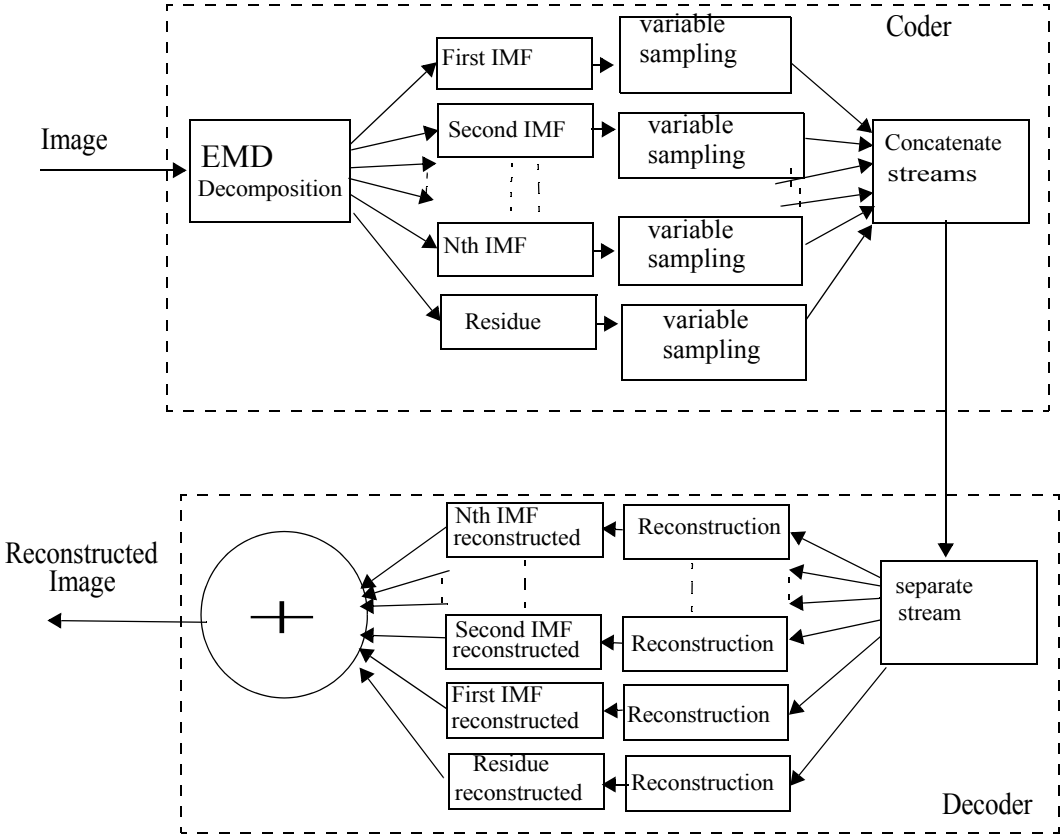


Figure 7.2. Variable sampling scheme.

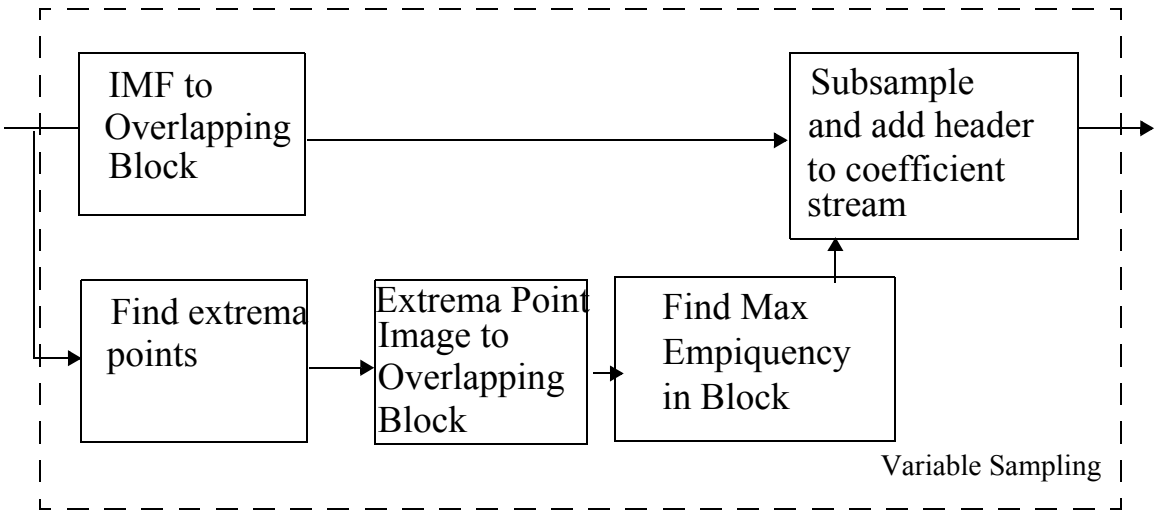


Figure 7.3. Details of the variable sampling block.

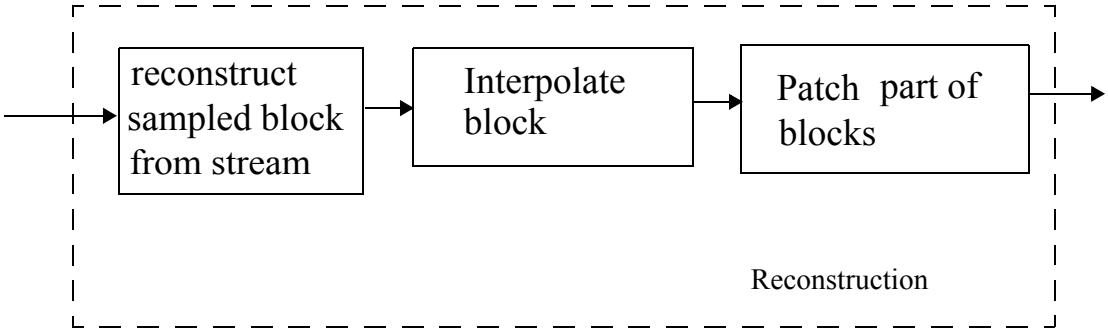


Figure 7.4. Details of the reconstruction block.

7.3 Results

The method of variable sampling is tested on the Lenna image of size 128 by 128 pixels, shown in Figure 7.13(a), using the EMD, shown in Figure 7.9 to the left. Other images are two different detail parts of the 512x512 Lenna image, one of size 64x64 pixels shown in Figure 7.11(a), using the EMD, shown in Figure 7.5 to the left, and the other one of size 128x128 pixels shown in Figure 7.12(a), using the EMD, shown in Figure 7.7 to the left. Each IMF is subsampled using the method of variable sampling with overlapping 7x7 blocks, as described in section 7.2. Reconstructing the subsampled blocks is done with cubic interpolation over a regular set of samples. The PSNR measure is computed from the original and the reconstructed image. The average sampling rate measure is a count of the number of samples used in relation to the total number of pixels in the image.

The results of the variable subsampling of the IMFs and the residue are presented in Figure 7.5 to Figure 7.10. To the left are the original IMFs, in the middle the reconstructed IMFs and to the right the parameters in terms of sampling rate and distortion. From this we can see that the first IMF needs more than a quarter of the pixels, every second sample in both directions, to be reconstructed at acceptable distortion. For the second and third IMF we see a behaviour which is more pleasing in that it has a signal-to-noise ratio of more than 36 dB, combined with a low sampling rate. For the fourth IMF and the residue we have reached the limit of subsampling which is one sample per block.

The image is reconstructed by the sum of reconstructed IMFs and residue. In Figure 7.11 to Figure 7.13 the original image is shown together with the reconstructed image. The sum of samples to represent all of the IMFs and the residue gives a value of the average sampling rate needed to represent the image.

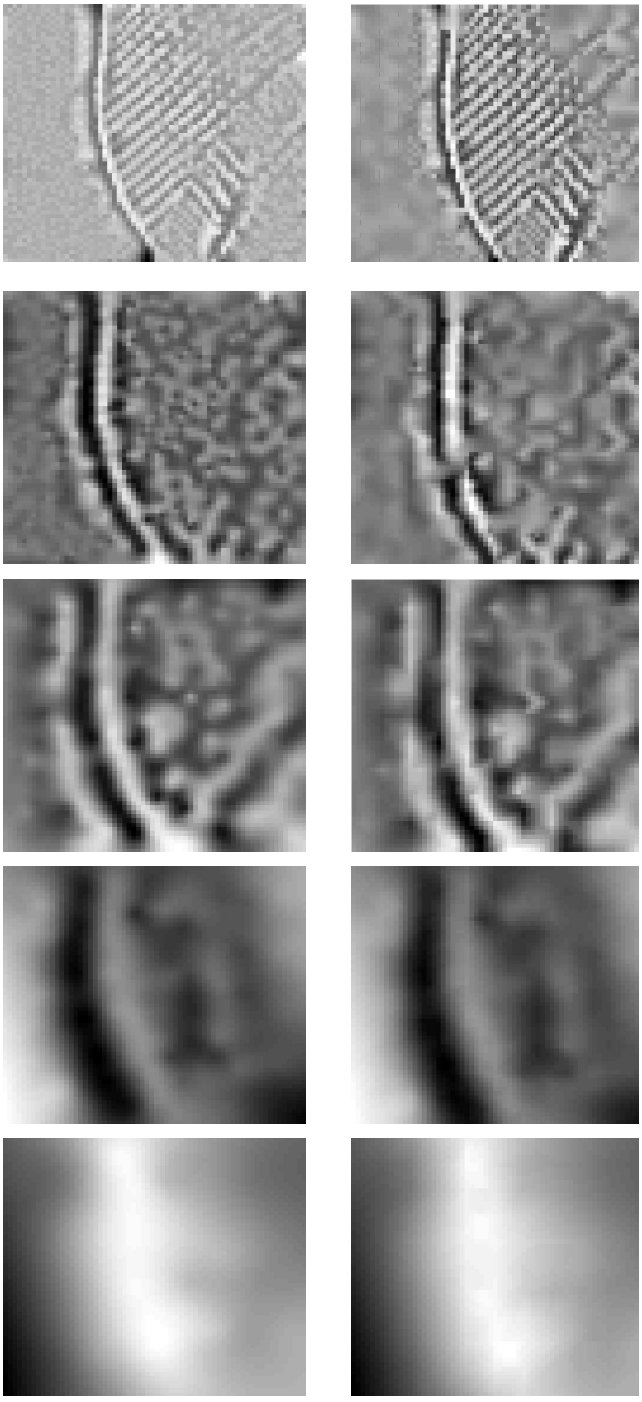


Figure 7.5. Detail one. The result of the variable subsampling of the set of IMFs and residue with zero zone set to 1. Left: original, Middle: reconstructed, Right: sampling rate and distortion.

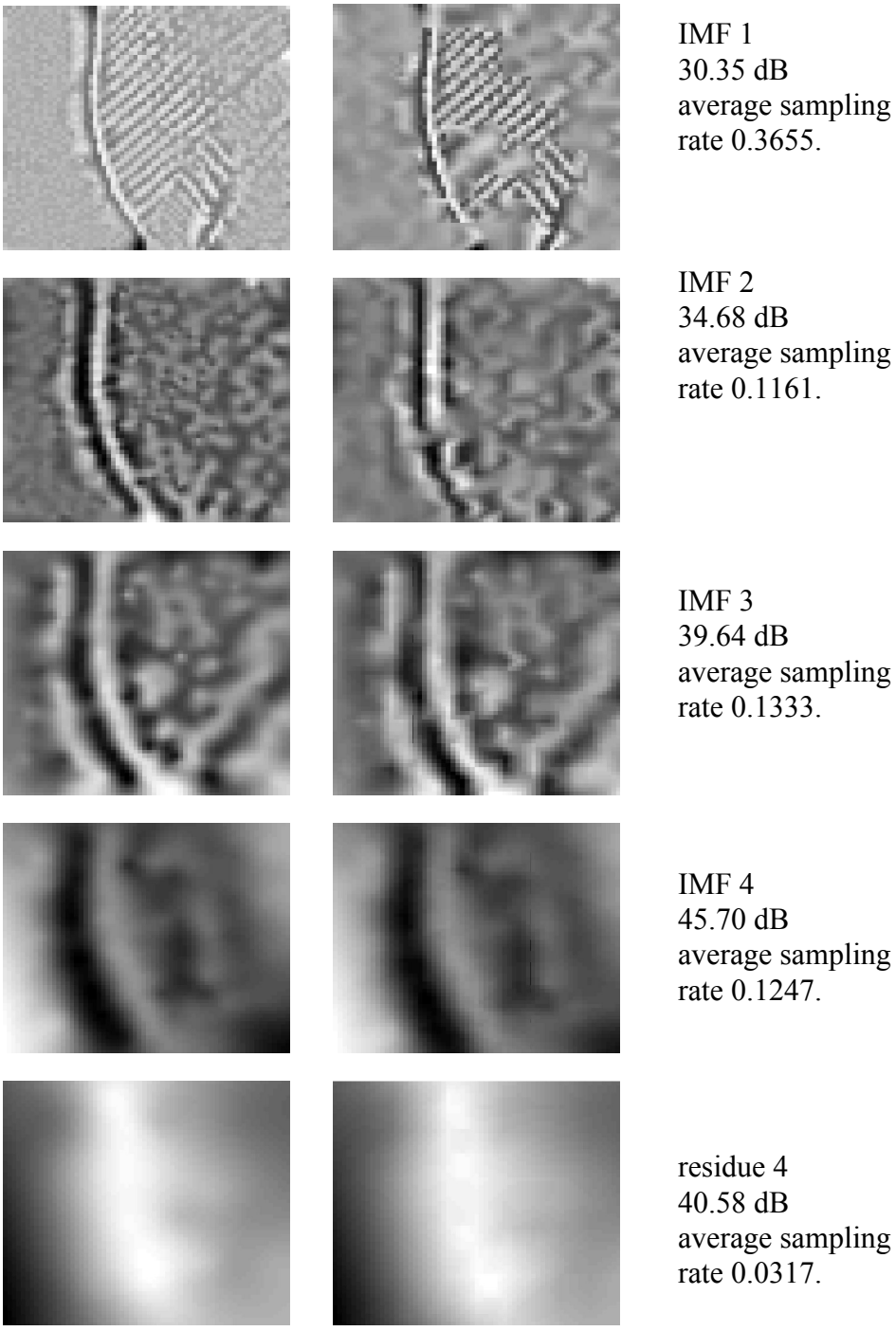


Figure 7.6. Detail one. The result of the variable subsampling of the set of IMFs and residue with zero zone set to 20. Left: original, Middle: reconstructed, Right: sampling rate and distortion.

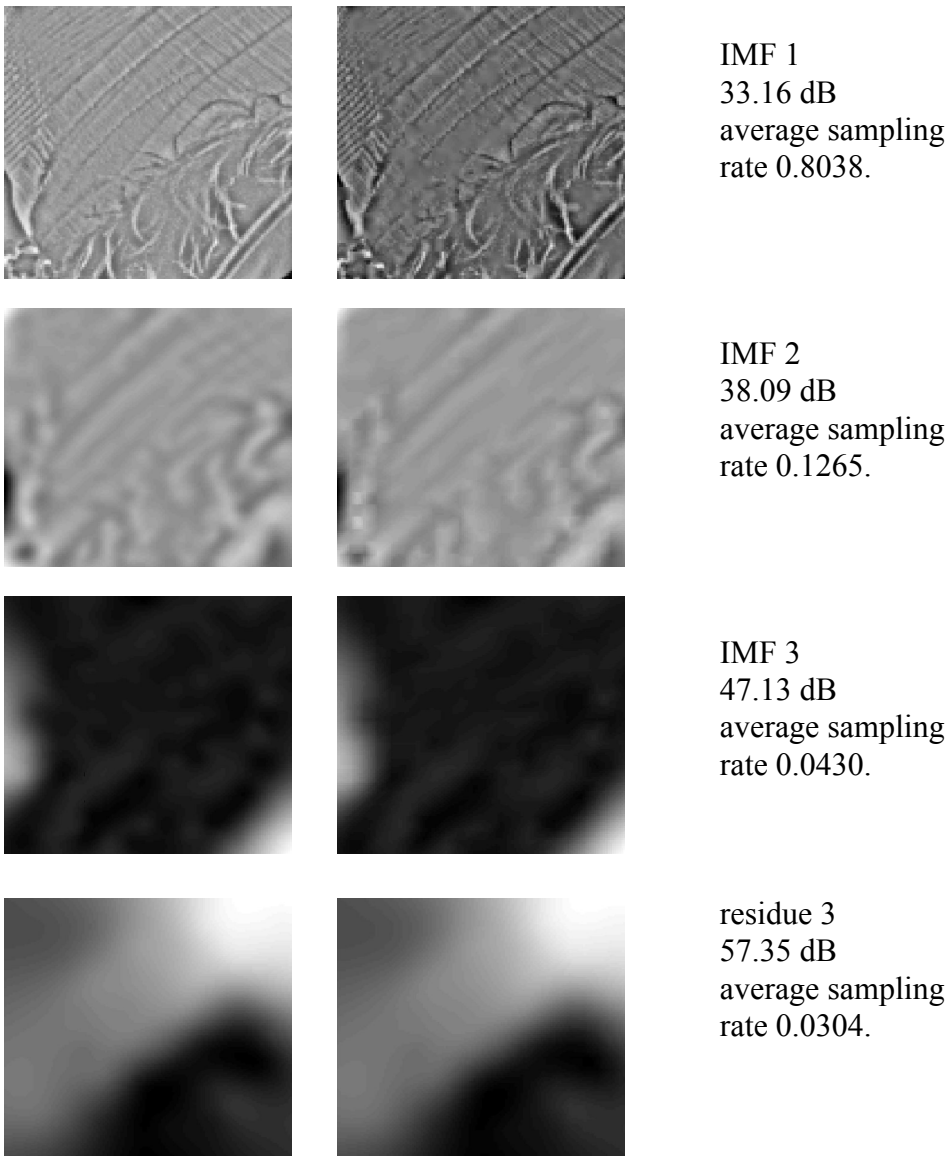
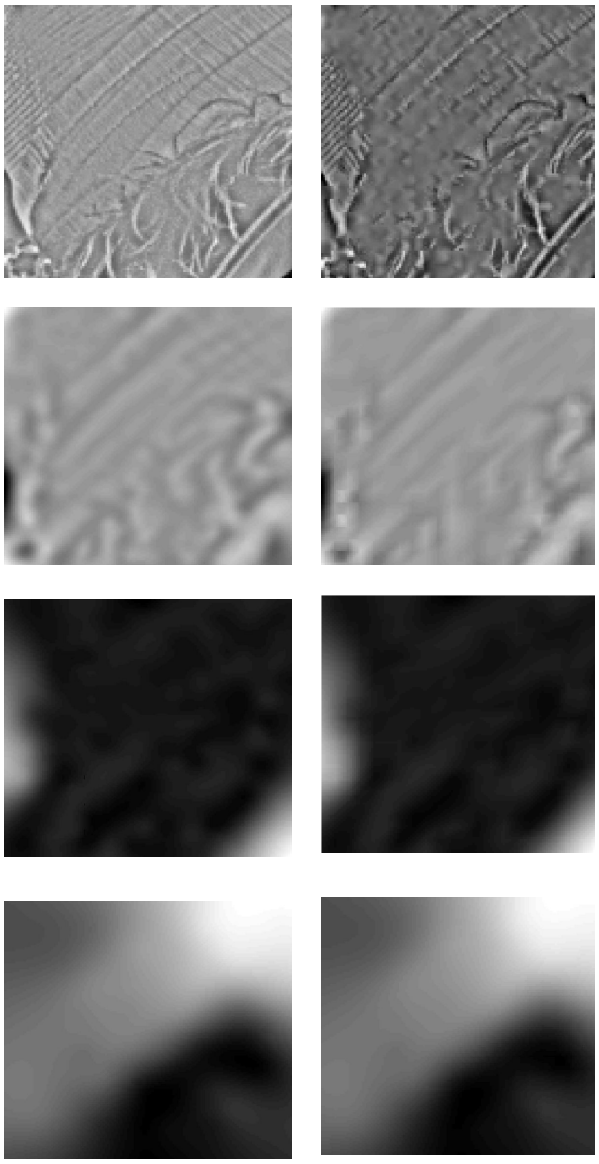


Figure 7.7. Detail two. The result of the variable subsampling of the set of IMFs and residue with zero zone set to 1. Left: original, Middle: reconstructed, Right: sampling rate and distortion.



IMF 1
29.47 dB
average sampling
rate 0.4367.

IMF 2
36.69 dB
average sampling
rate 0.0578.

IMF 3
46.73 dB
average sampling
rate 0.0273.

residue 3
57.35 dB
average sampling
rate 0.0304.

Figure 7.8. Detail one. The result of the variable subsampling of the set of IMFs and residue with zero zone set to 20. Left: original, Middle: reconstructed, Right: sampling rate and distortion.

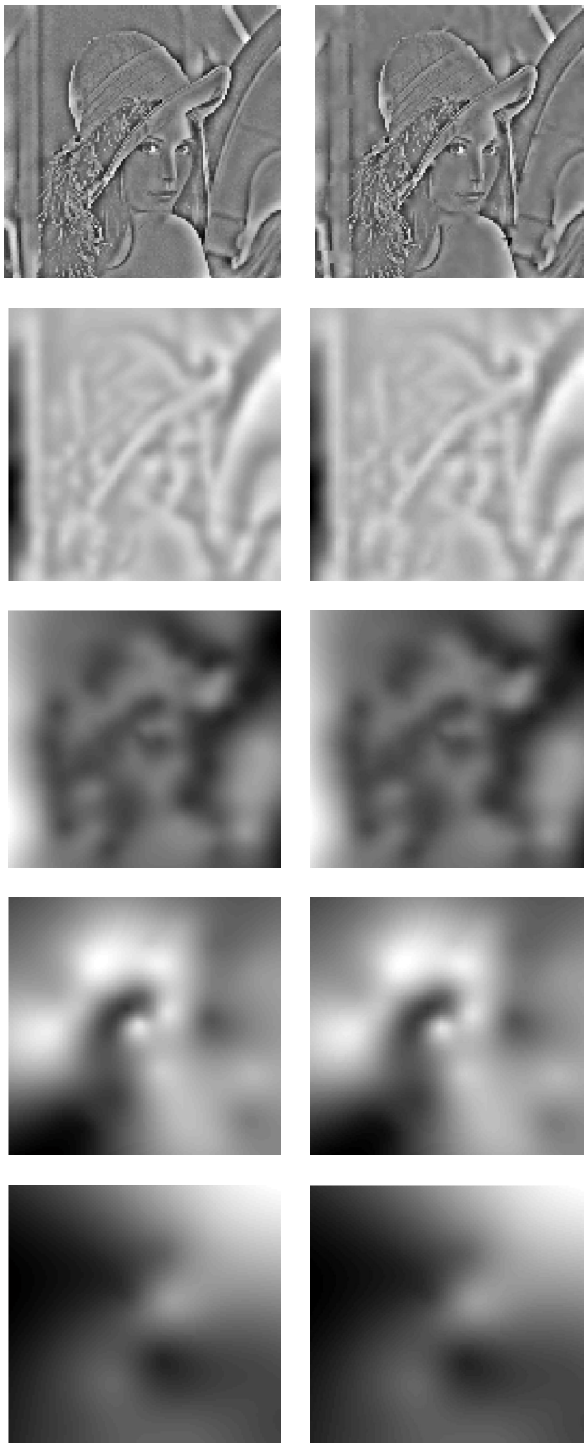


Figure 7.9. Lenna 128. The result of the variable subsampling of the set of IMFs and residue with zero zone set to 1. Left: original, Middle: reconstructed, Right: sampling rate and distortion.

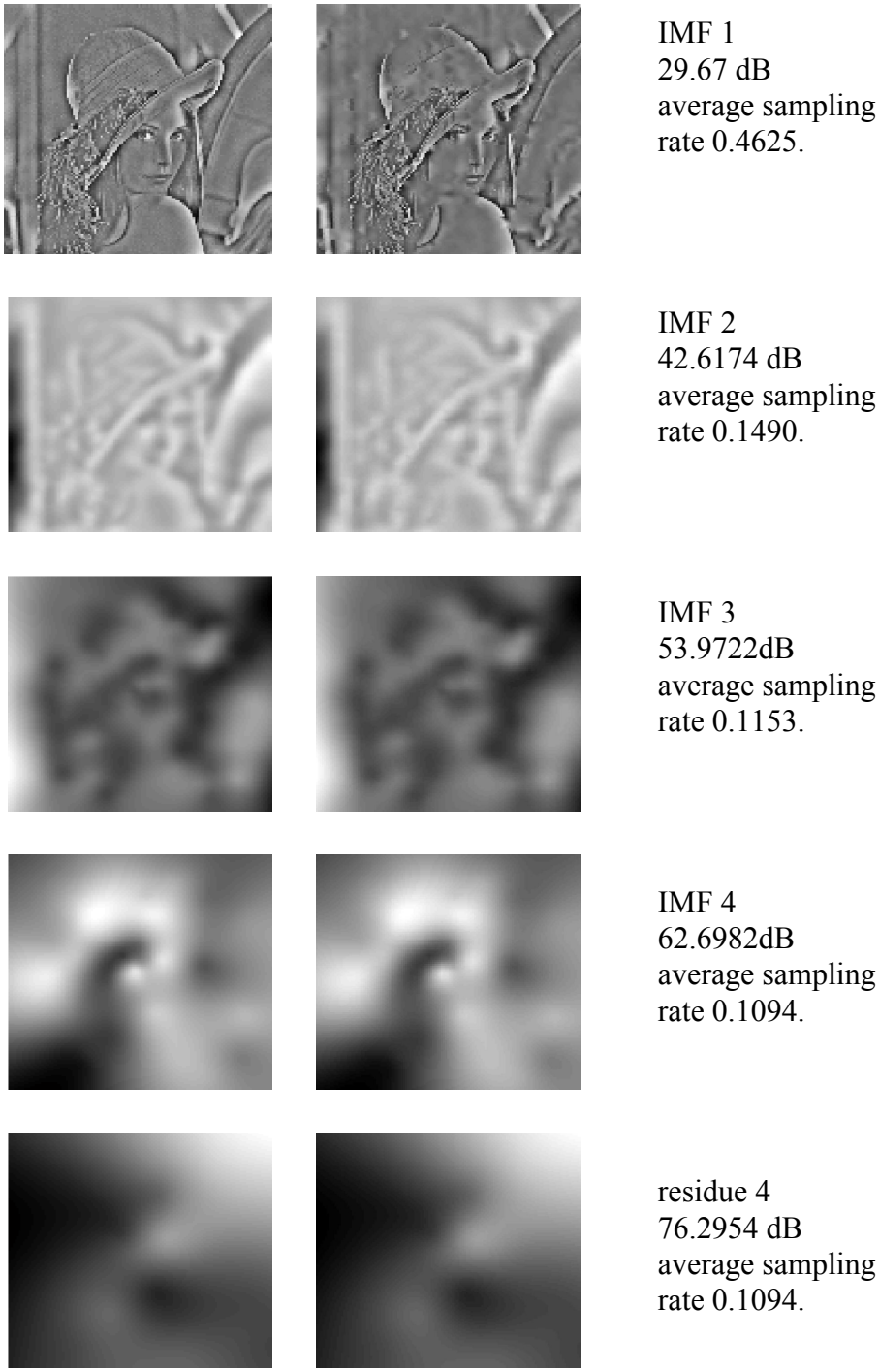
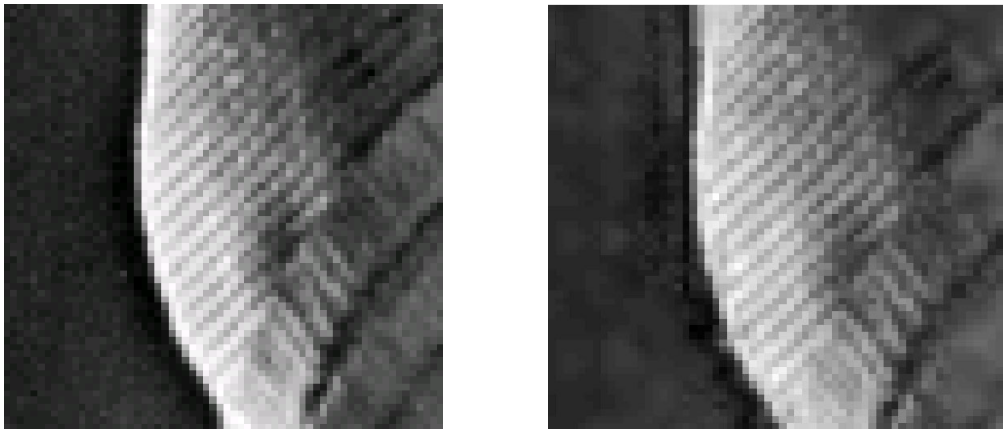
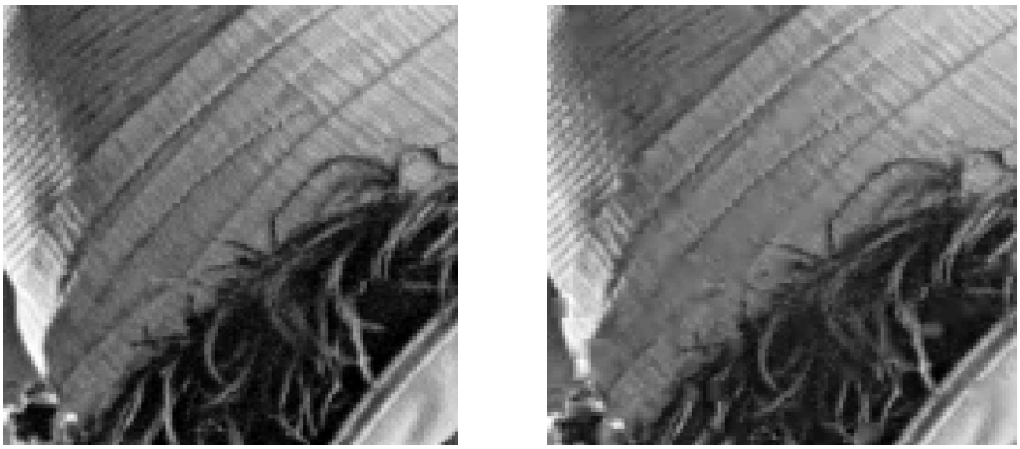


Figure 7.10. Lenna 128. The result of the variable subsampling of the set of IMFs and residue with zero zone set to 20. Left: original, Middle: reconstructed, Right: sampling rate and distortion.



a b

Figure 7.11. Detail one. a) Original image, b) Reconstructed image by the summation of reconstructed IMFs and residue, 30.45 dB, average sampling rate 1.0637



a b

Figure 7.12. Detail two. a) Original image, b) Reconstructed image by the summation of reconstructed IMFs and residue, 30.9934 dB, average sampling rate 0.9193.

a b

Figure 7.13. Lenna 128. a) Original image, b) Reconstructed image by the sum of reconstructed IMFs and residue, 31.8 dB using an average sampling rate of 0.86.

7.4 Summary

We have proposed a method for variable sampling of the two-dimensional EMD. This is done block wise using the non-uniformly located extrema points of the IMF to control the uniform sampling rate of the block. The very special properties of the IMF can be used for variable sampling of the IMFs and the residue in order to reduce the number of parameters to represent the image. The empiquency varies in the IMF and thus can be used to control the sampling rate adaptive over the image. Using overlapping blocks reduces blocking artifacts.

A method of using overlapping 7x7 blocks is introduced to overcome blocking artifacts and to further reduce the number of parameters to represent the image. The results presented here show that subsampling offers a way to keep the total numbers of samples generated by EMD approximately equal to the number of pixels of the original image.

Chapter 8

Entropy Coding of Variable Sampling Components

The variable sampling in Chapter 7 presents a promising method to treat the EMD. The sampling process leaves us with a number of samples for each block. In this chapter we will concentrate on entropy coding the samples from variable sampling.

8.1 Coder outline

This approach uses almost the same scheme as the variable sampling. The difference is that after the subsampling the samples are entropy coded using Huffman coding or fix-length coding of the samples. The coder is shown in Figure 8.1.

The details of the variable sampling block are shown in Figure 7.3. The IMF is searched for extrema points and both the extrema point image and the IMF itself are divided into overlapping blocks. The maximum empiquency found in the extrema point image block determines the sampling rate for the subsampling of the corresponding IMF block. The samples are represented by one sample alone or 6x6, 3x3, or 2x2 blocks of samples. The whole set of samples is entropy coded and the components are quantized before the two-bit block header is added to each block.

The details of the reconstruction block are shown in Figure 7.4. The two-bit header indicates the sampling rate. With this information the components achieved from the stream can be placed in their proper place in the 7x7 pixel block. The stream only contains samples for the 6x6 block, the missing samples are found in the reconstructed neighbouring blocks. Since the missing samples are located in the rightmost column and the lowest row of the block, the reconstruction starts with the block in the lower right corner working row-wise through the blocks. For the blocks with no neighbours holding missing samples

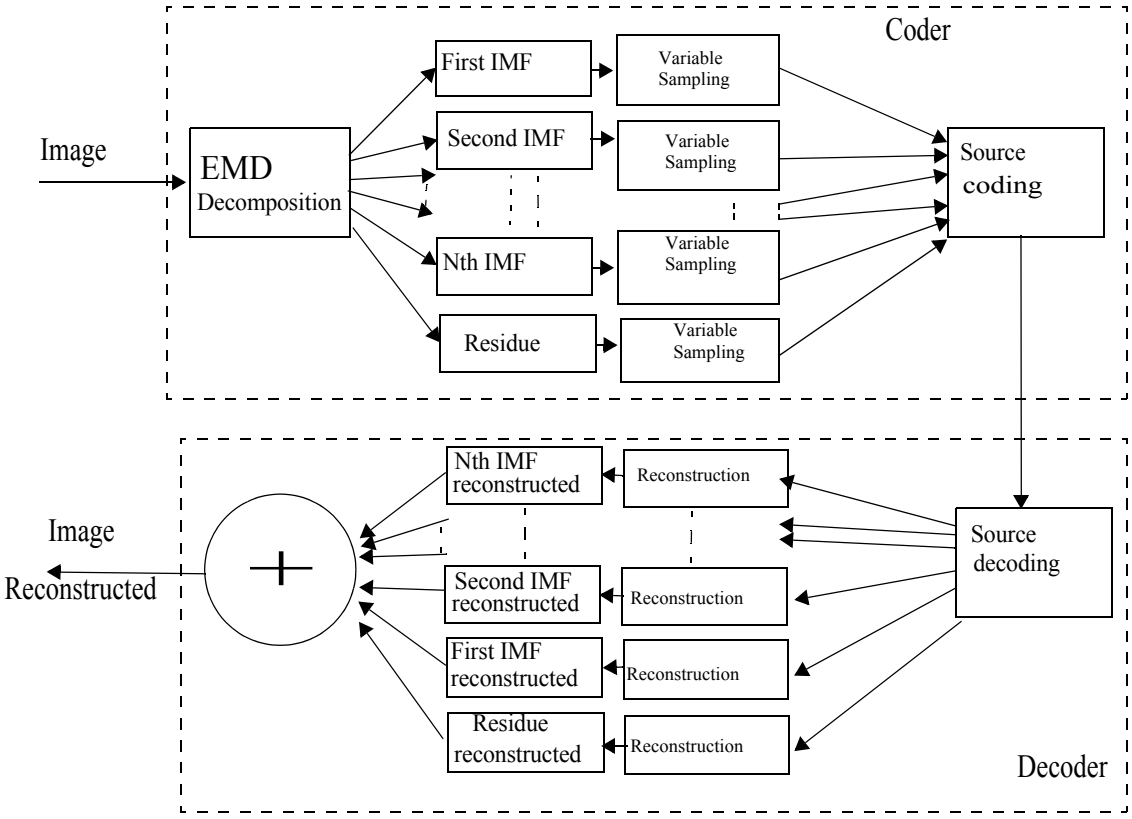


Figure 8.1. Variable sampling entropy coding scheme.

dummy samples are used. The 7x7 blocks of samples is interpolated and the nonoverlapping 6x6 part of the reconstructed block is used to generate the output image.

8.1.1 Results

The coder is tested on the EMD of three different images, Detail one, Detail two, and Lenna128, using zero zone 1, 10, and 20. The quantization varies from 256 levels down to 4 levels. Figure 8.2, Figure 8.4, and Figure 8.6 show the coding results for the EMD of the images, respectively. In Figure 8.3, Figure 8.5, and Figure 8.7, example reconstructions of the IMFs and residue using fine quantization are presented along with the original image and the parameters used. The coding by the Huffman method gives better performance than the fixlength coding method for the first IMF while for the rest of the IMFs and the residue the fixlength coding method gives better performance than the Huffman method. Figure 8.8, Figure 8.9, and Figure 8.9 show the images reconstructed using fine quantization.

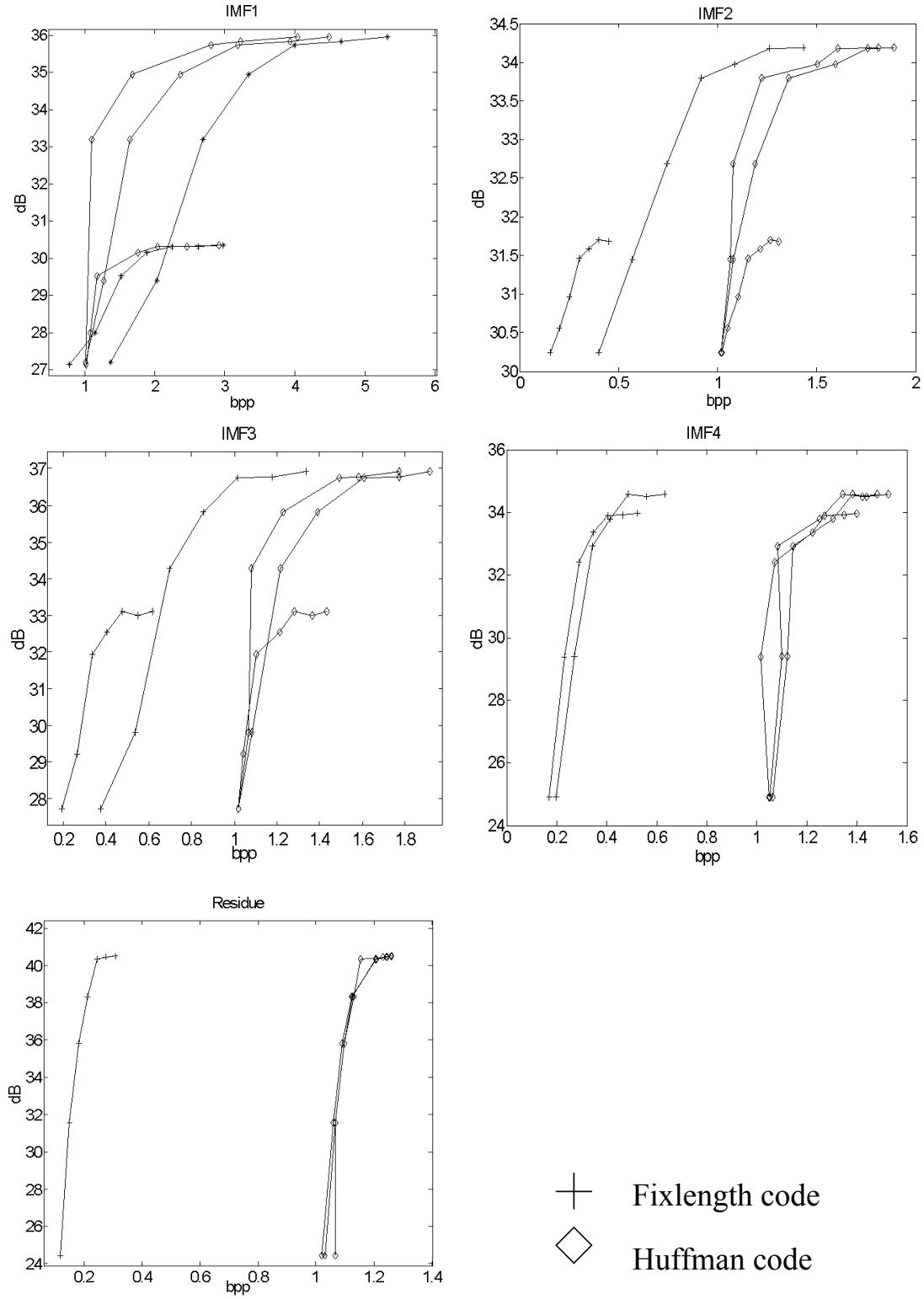


Figure 8.2. Entropy coding result of Detail one. Zero zone = 1, 10, 20 quantization varies from 256 levels down to 4 levels.

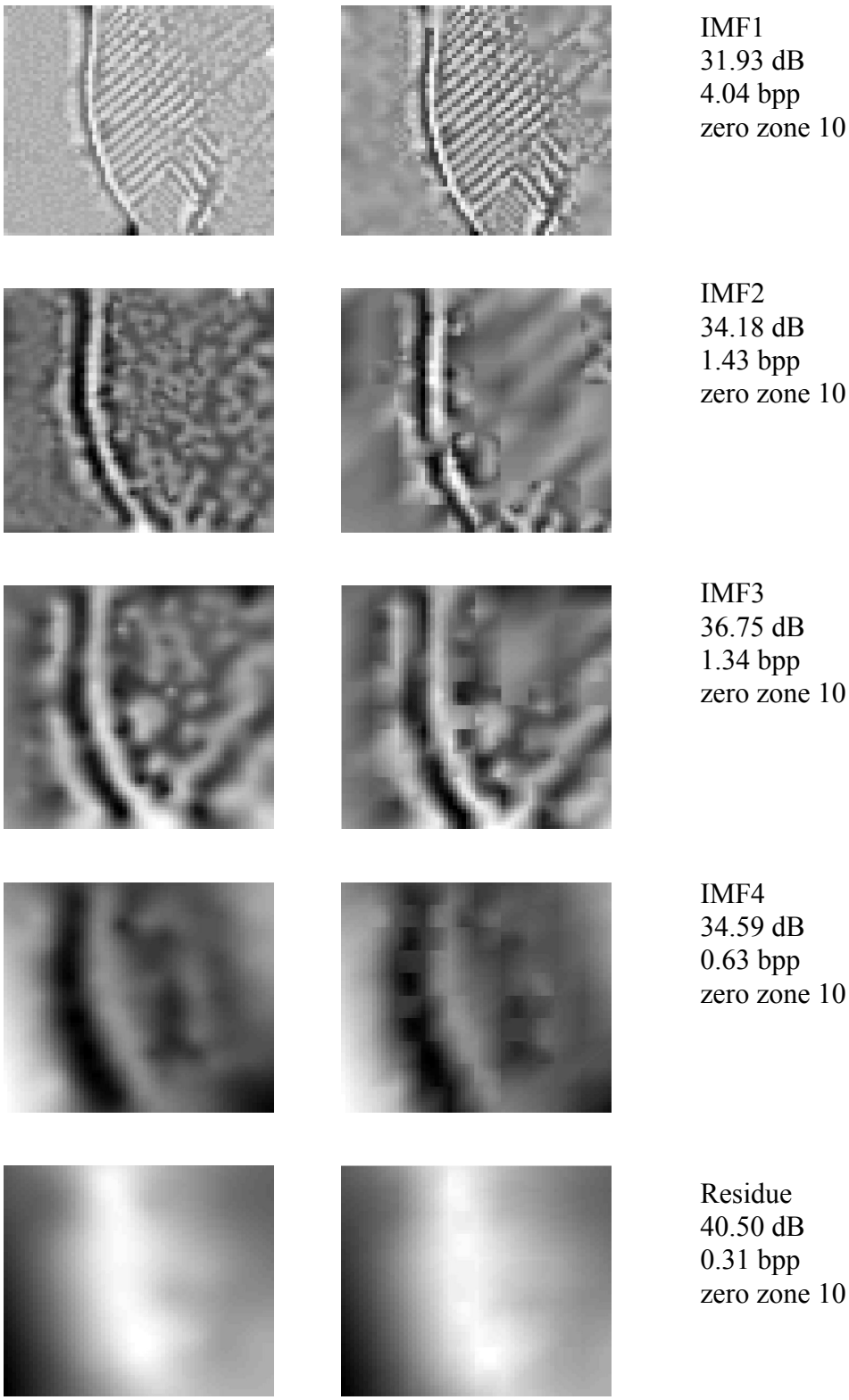


Figure 8.3. The result of the variable subsampling entropy coding of the set of IMF and residue for Detail one using fine quantization. Original to left, reconstructed in the middle, the parameters specified to the right.

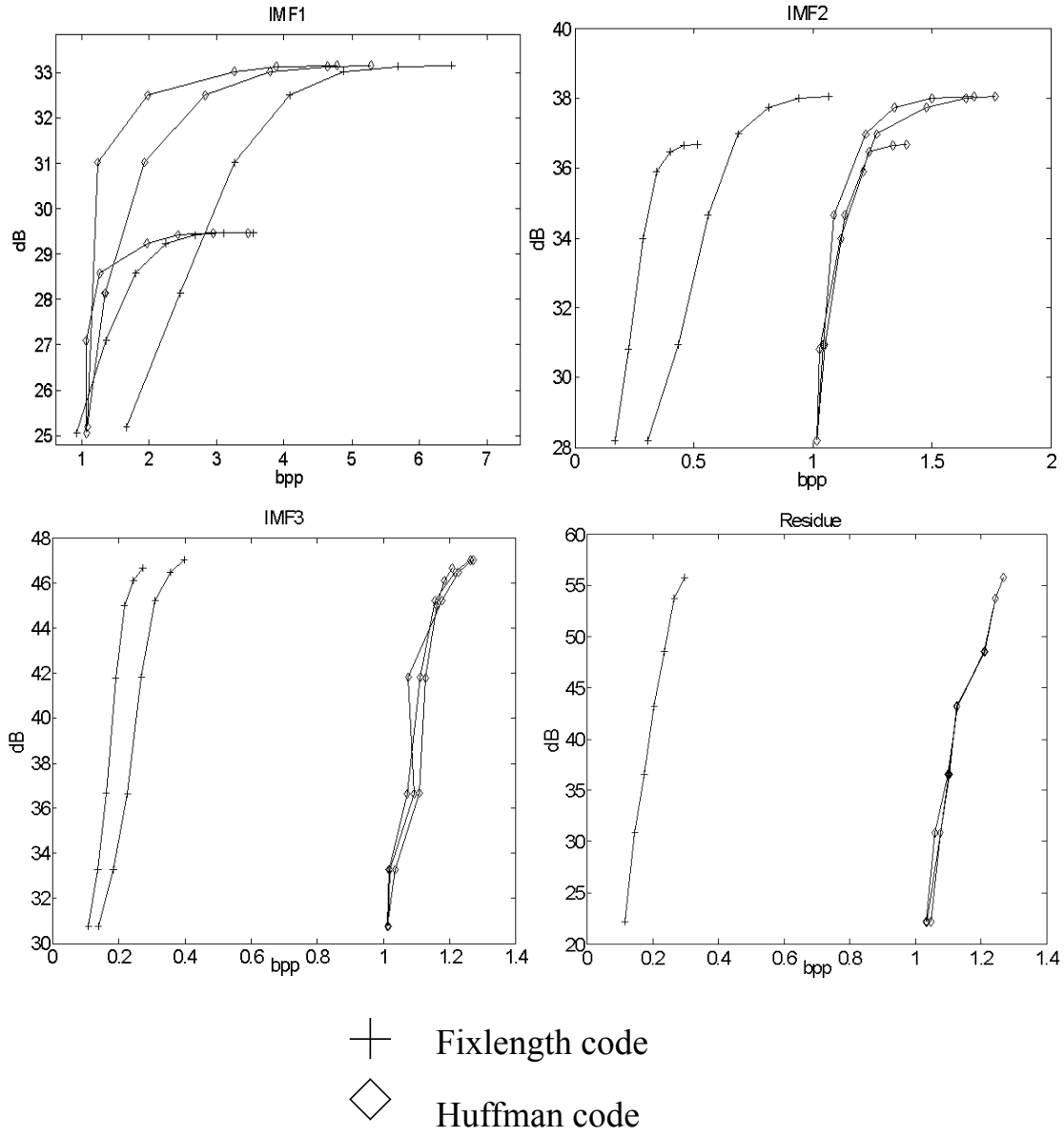


Figure 8.4. Entropy coding result of Detail two. Zero zone=1, 10, 20 quantization varies from 256 levels down to 4 levels.

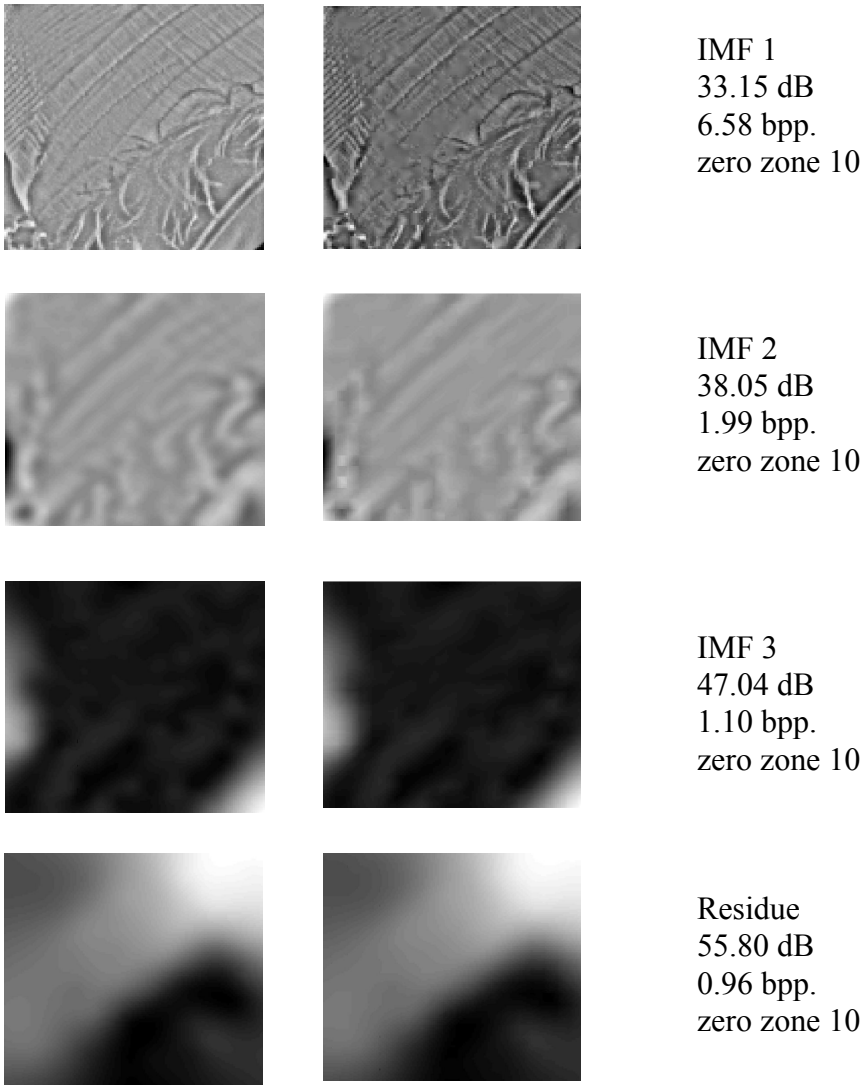


Figure 8.5. The result of the variable subsampling entropy coding of the set of IMF and residue for Detail two with different values on the zero zone and fine quantization. Left: original, Middle: reconstructed, Right: bitrate and distortion.

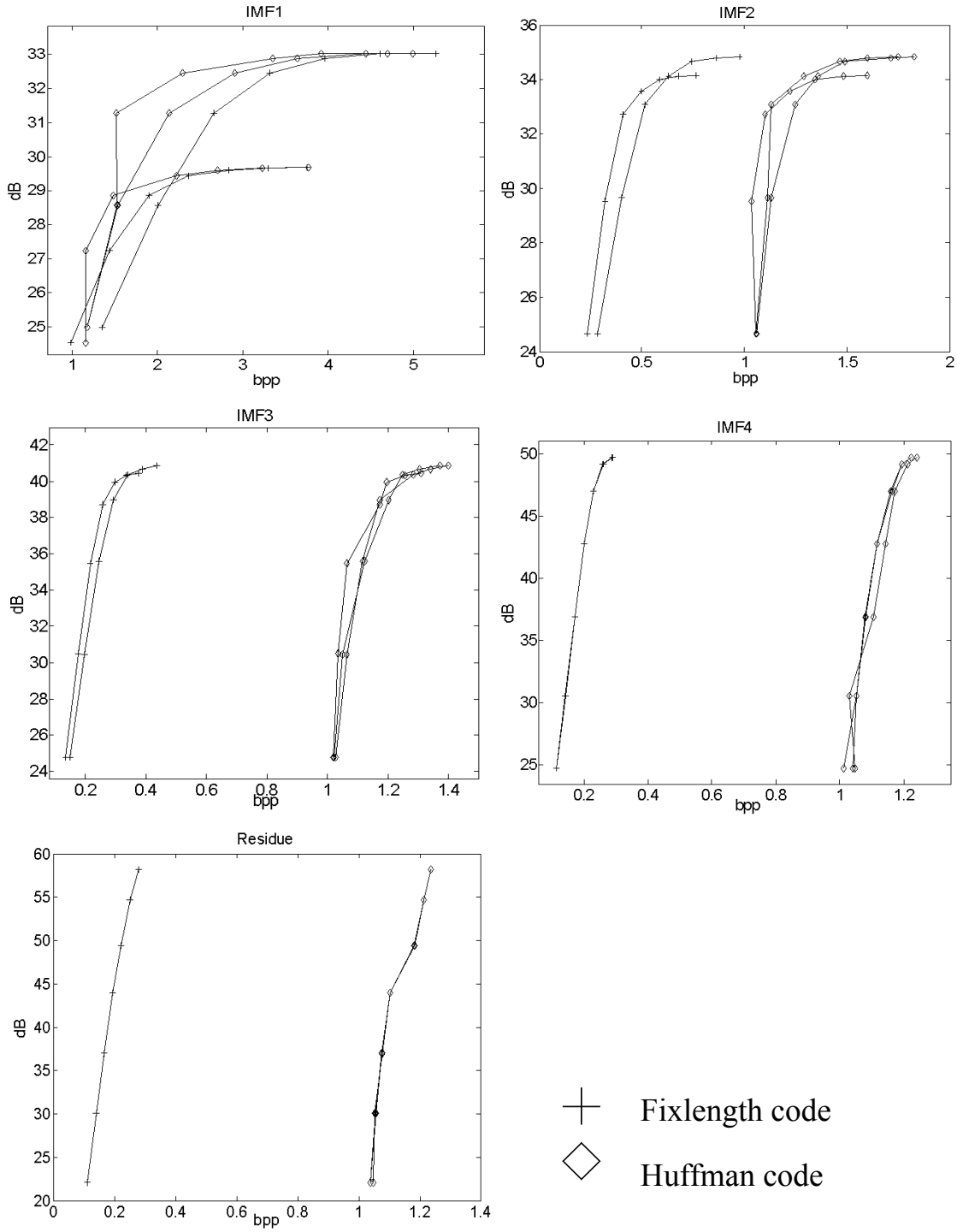


Figure 8.6. Entry coding result of Lenna128. Zero zone=1, 10, 20 quantization varies from 256 levels down to 4 levels.

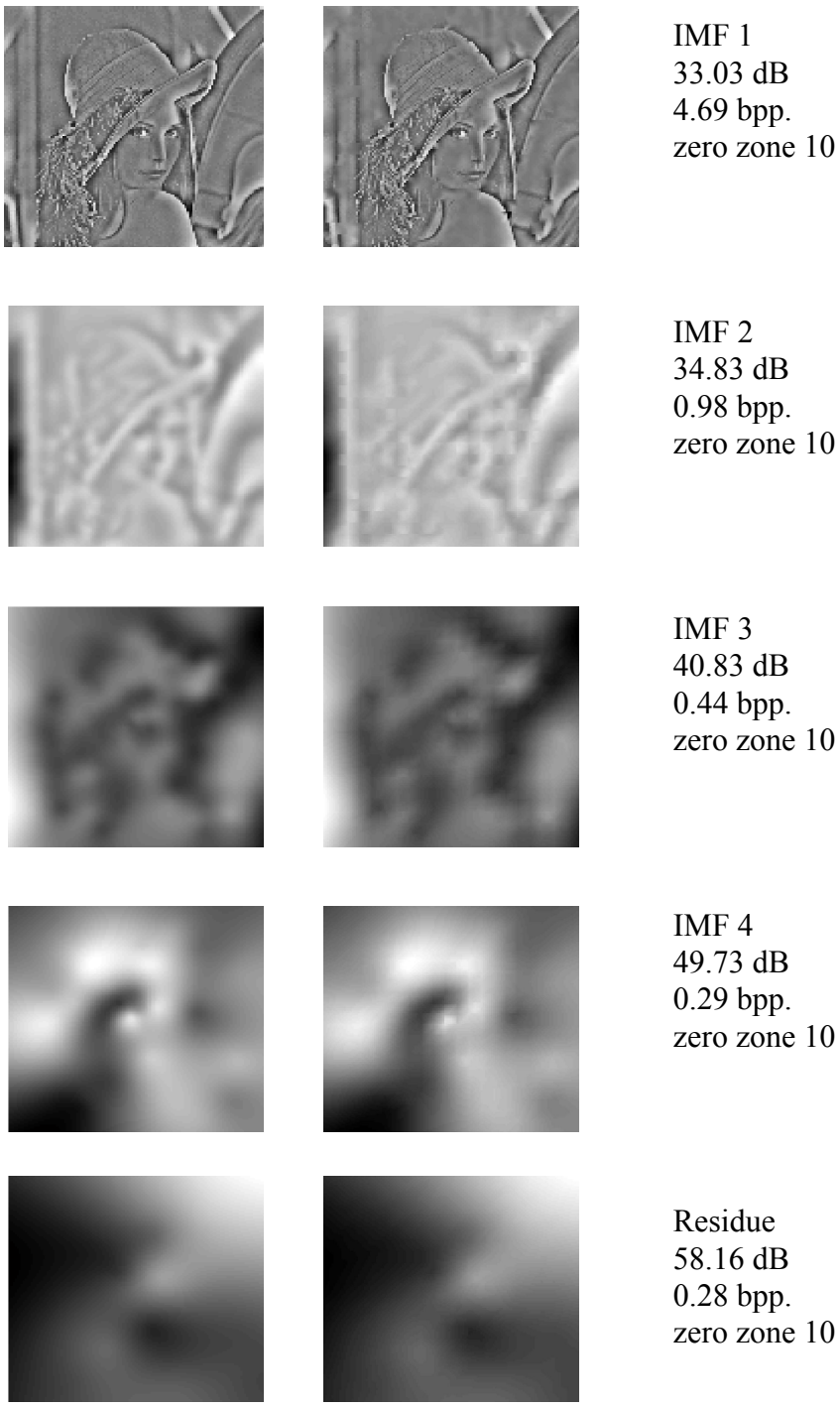
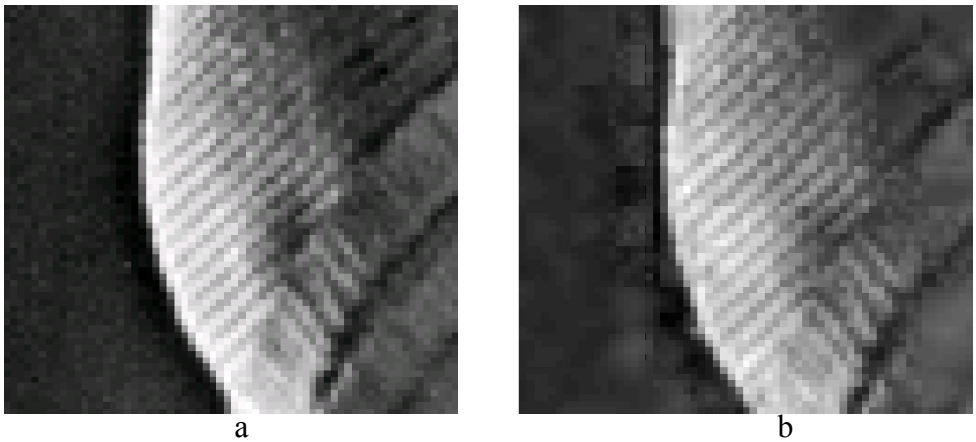
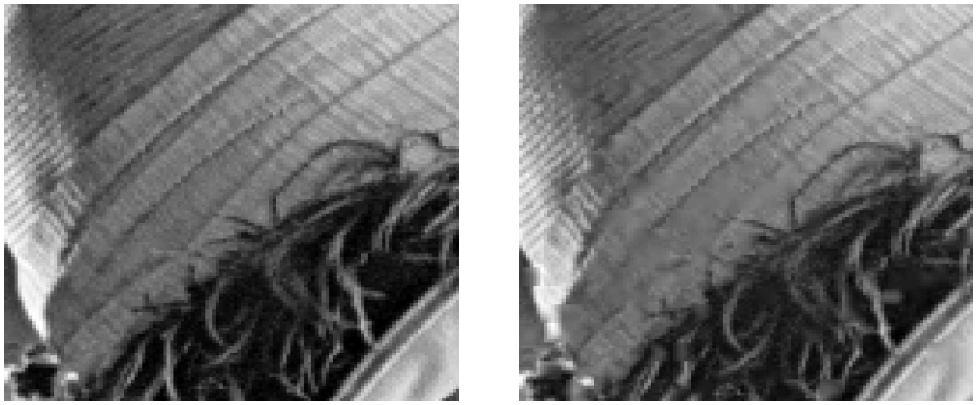


Figure 8.7. The result of the variable subsampling entropy coding of the set of IMF and residue for Lenna128 with zero zone set to 10 and fine quantization. Left: original, Middle: reconstructed, Right: sampling rate and distortion.



a b

Figure 8.8. a) Detail one original, b) Detail one reconstructed to 28.49 dB using 7.47 bpp. Fine quantization.



a b

Figure 8.9. a) Detail two original, b). Detail two image reconstructed to 31.41 dB using 6.55 bpp. Fine quantization.

a b

Figure 8.10. a) Lenna128 original, b) Lenna128 image reconstructed to 30.15 dB using 6.67 bpp. Fine quantization.

Chapter 9

EMD Image Coding using Variable Sampling of Overlapping Blocks

The entropy coding of the variable sampling presented in Chapter 8 shows the compression potential of the EMD. In this chapter we will concentrate on using the variable sampling method for compression. We start with coding the samples of the IMF and residue with the DCT. We then try coding the image by using the DCT of the variable sampled blocks for the first IMF and coding the first residue using the threshold DCT method presented in Chapter 6.

9.1 Coding of the EMD using DCT of the variable sampled blocks (VSDCTEMD)

The structure of the variable sampling described in Chapter 7 is inherited in this coding approach. The sampling process leaves us with a number of samples for each block. These can be squeezed into blocks of smaller size which can be DCT coded. The DCT components are then quantized and thresholded, leaving us with even fewer components to represent the block.

9.1.1 Coder outline

This approach uses almost the same scheme as the variable sampling. The difference is

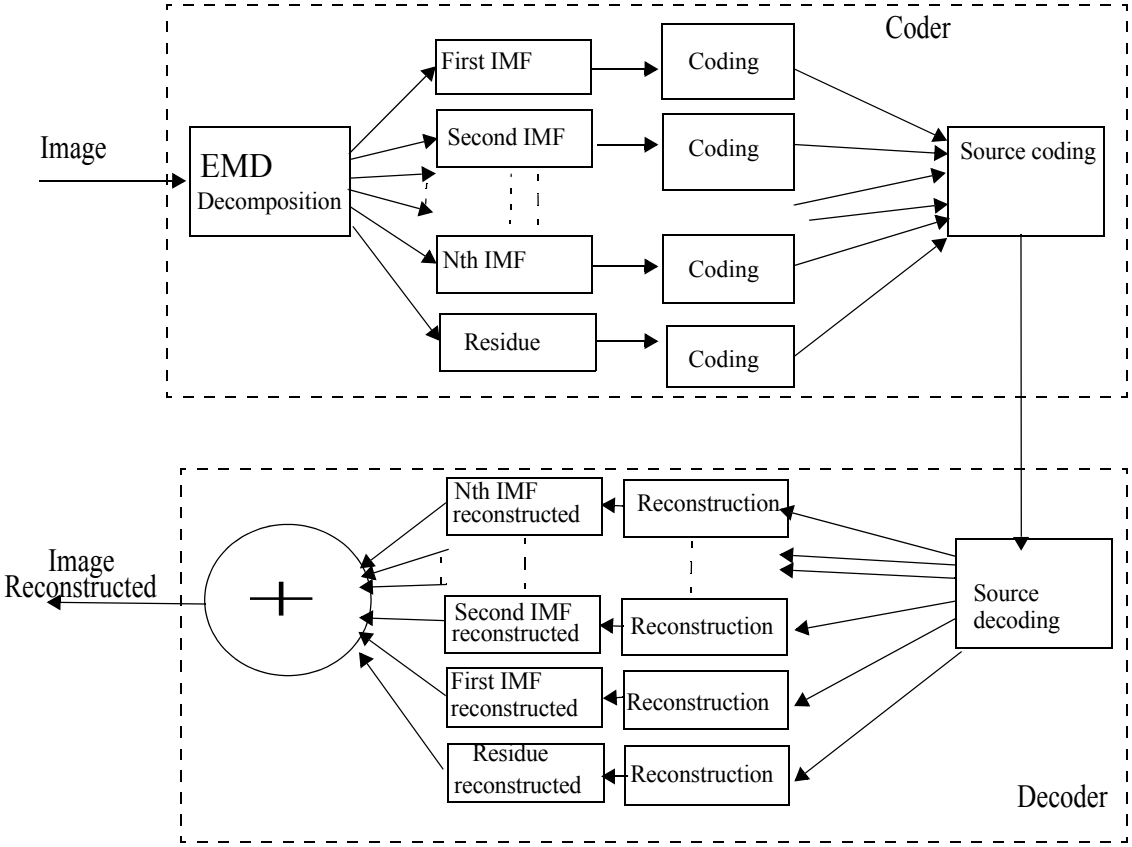


Figure 9.1. Variable sampling DCT coding scheme (VSDCTEMD).

that after the subsampling the samples are coded using DCT. The coder is shown in Figure 9.1

The details of the coding block are shown in Figure 9.2. the imf is searched for extrema points and both the extrema point image and the IMF itself are divided into overlapping blocks. The maximum empiquency found in the extrema point image block determines the sampling rate for the subsampling of the corresponding IMF block. The samples are represented by one sample alone or 6x6, 3x3, or 2x2 blocks of samples. These are DCT coded and the components are quantized and thresholded before the two-bit block header is added. The resulting component stream is Huffman coded or fixlength coded.

The details of the reconstruction block are shown in Figure 9.3. The two-bit header indicates the sampling rate. With this information the components achieved from the inverse DCT transform of the stream can be placed in their proper place in the 7x7 pixel block. The stream only contains samples for the 6x6 block, the missing samples are found in the reconstructed neighbouring blocks. Since the missing samples are located in the rightmost column and the lowest row of the block, the reconstruction starts with the block in the lower right corner working row-wise through the blocks. For the blocks with no neighbours holding missing samples dummy samples are used. The 7x7 blocks of samples are interpolated and the nonoverlapping 6x6 part of the reconstructed block is used to generate the output image.

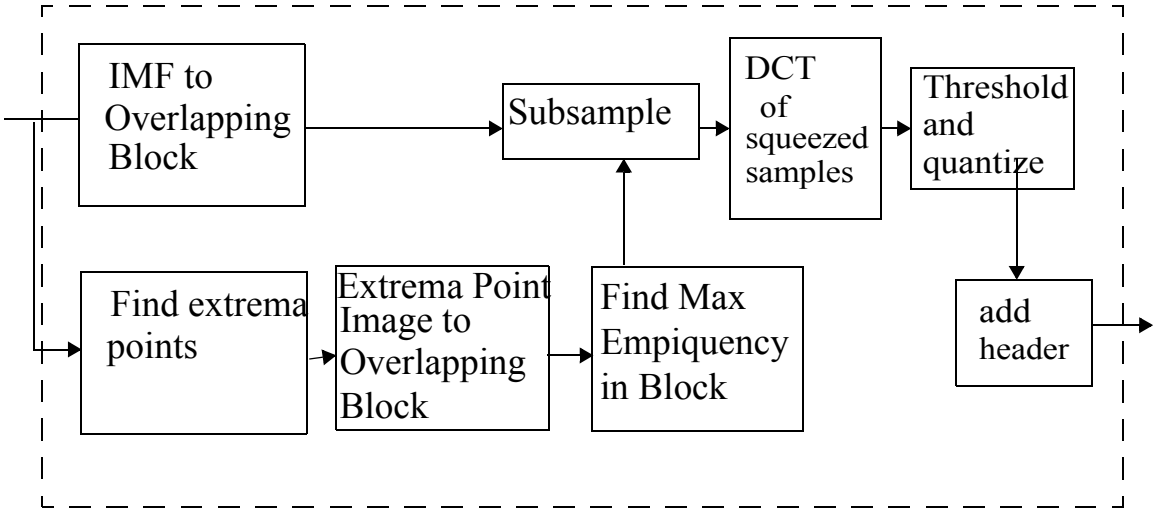


Figure 9.2. Details of the coding block.

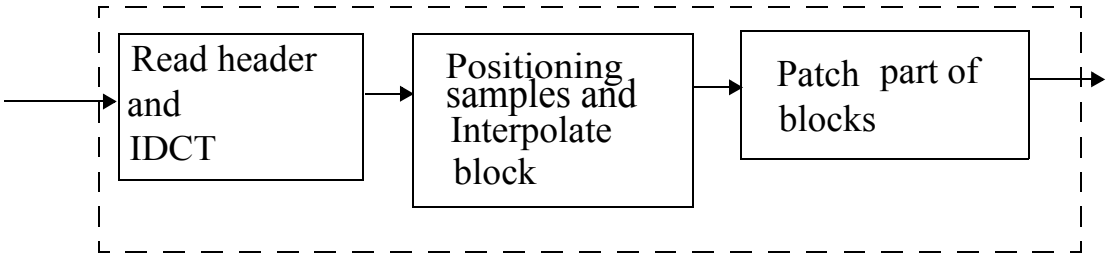


Figure 9.3. Details of the reconstruction block.

9.1.2 Results

The VSDCTEMD coder is tested on three different images, Detail one, Detail two, and Lenna128, using zero zones 1, 10, and 20. The quantization varies from 256 levels down to 8 levels. The DCT components are thresholded leaving only those having a value larger than or equal to the threshold. The threshold varies from 0.1% to 100% of the maximum value. Figure 9.4, Figure 9.6, and Figure 9.8, show the coding results for the EMD of the images, respectively, and in Figure 9.5, Figure 9.7, and Figure 9.9, example reconstructions of the IMFs and residue are presented along with the original image and the parameters used. The coding method gives good results for all IMFs and the residue. Even for the difficult first IMF the result is over 30 dB for a bitrate of 1bpp. Example reconstructions are presented in Figure 9.10, Figure 9.11 and Figure 9.12, along with the original image.

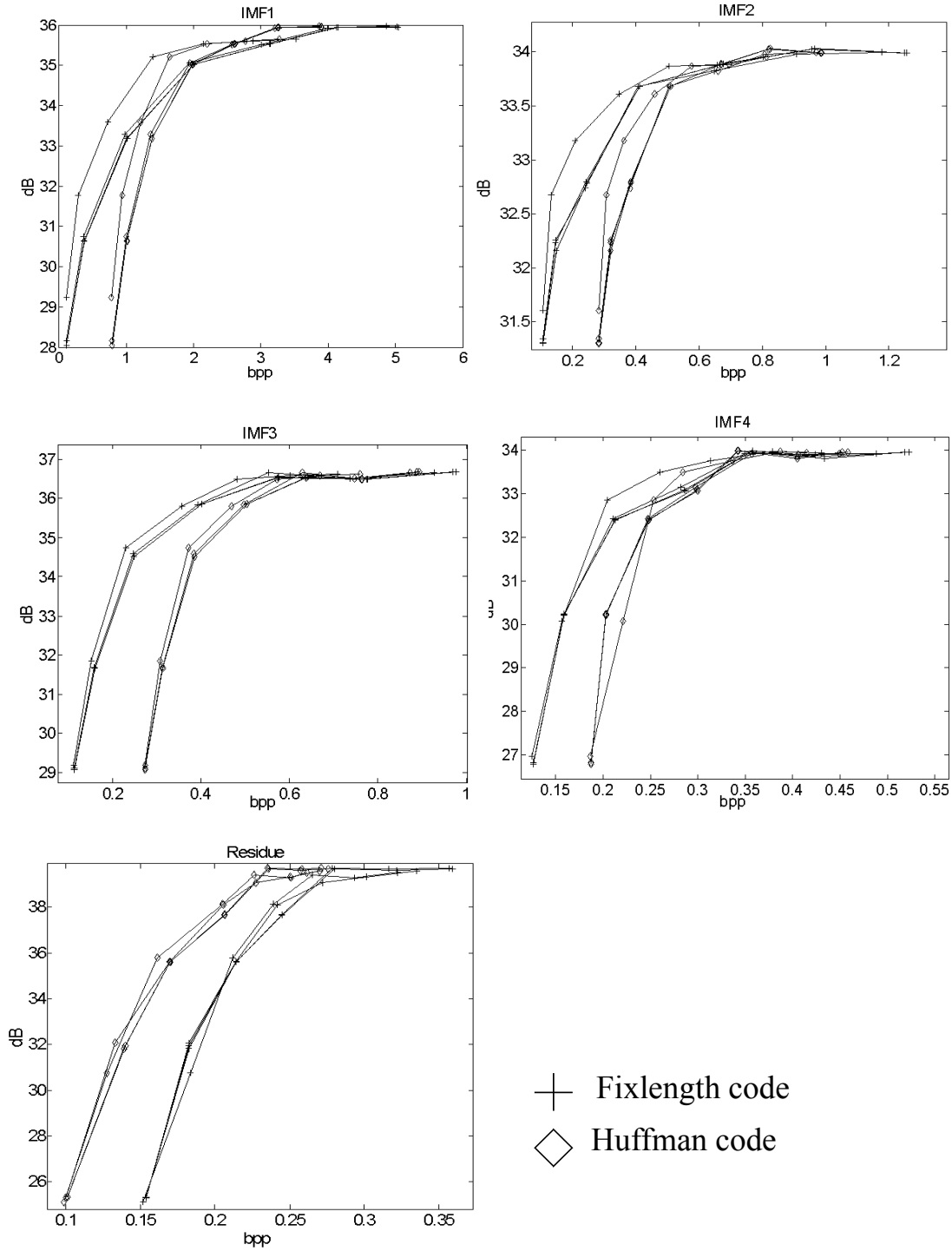


Figure 9.4. VSDCTEMD Coding results of Detail one. Zero zone=1, 10, 20 quantization varies from 256 levels down to 8 levels.

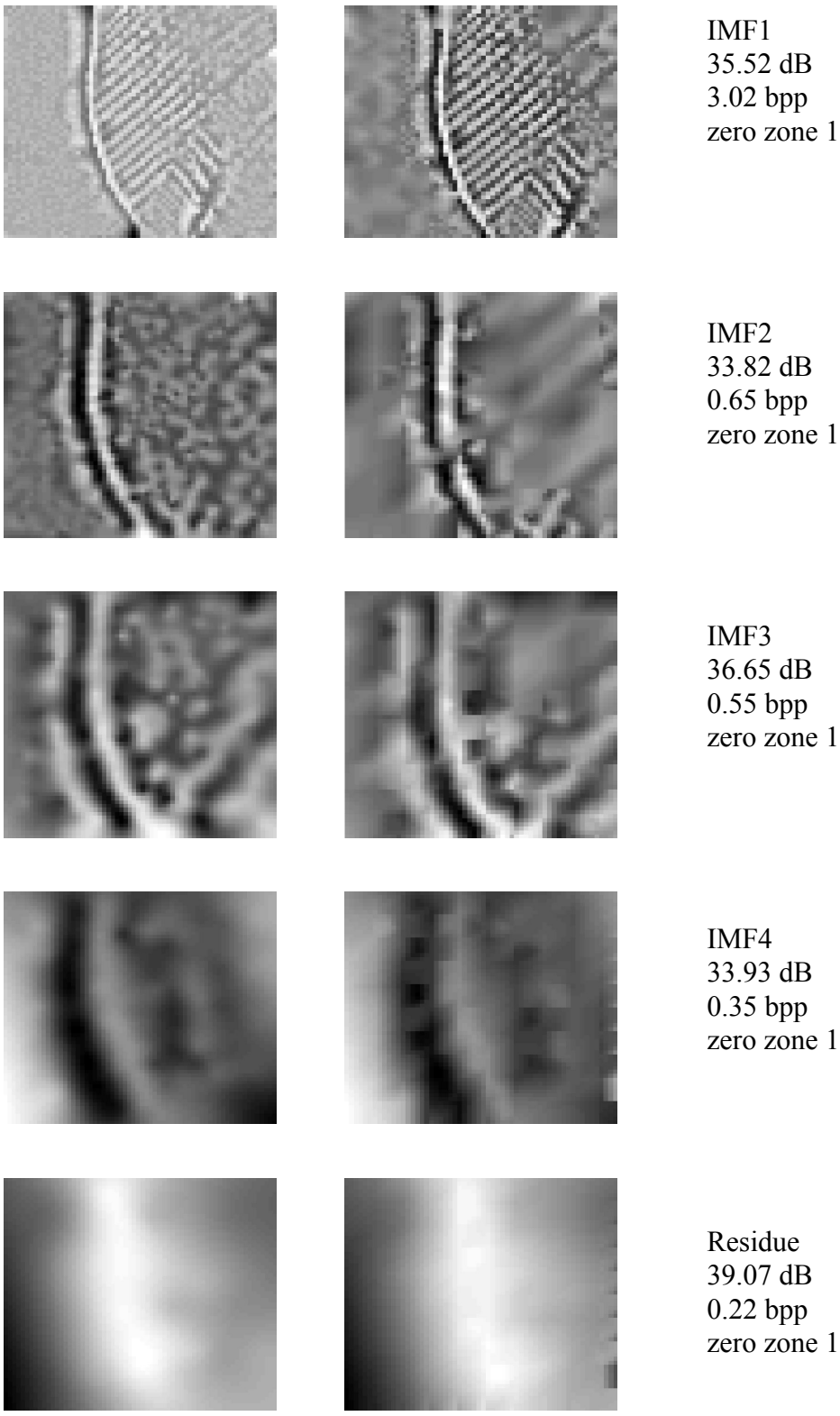


Figure 9.5. The result of VSDCTEMD coding of Detail one. Original to left, reconstructed in the middle, using the parameters specified to the right.

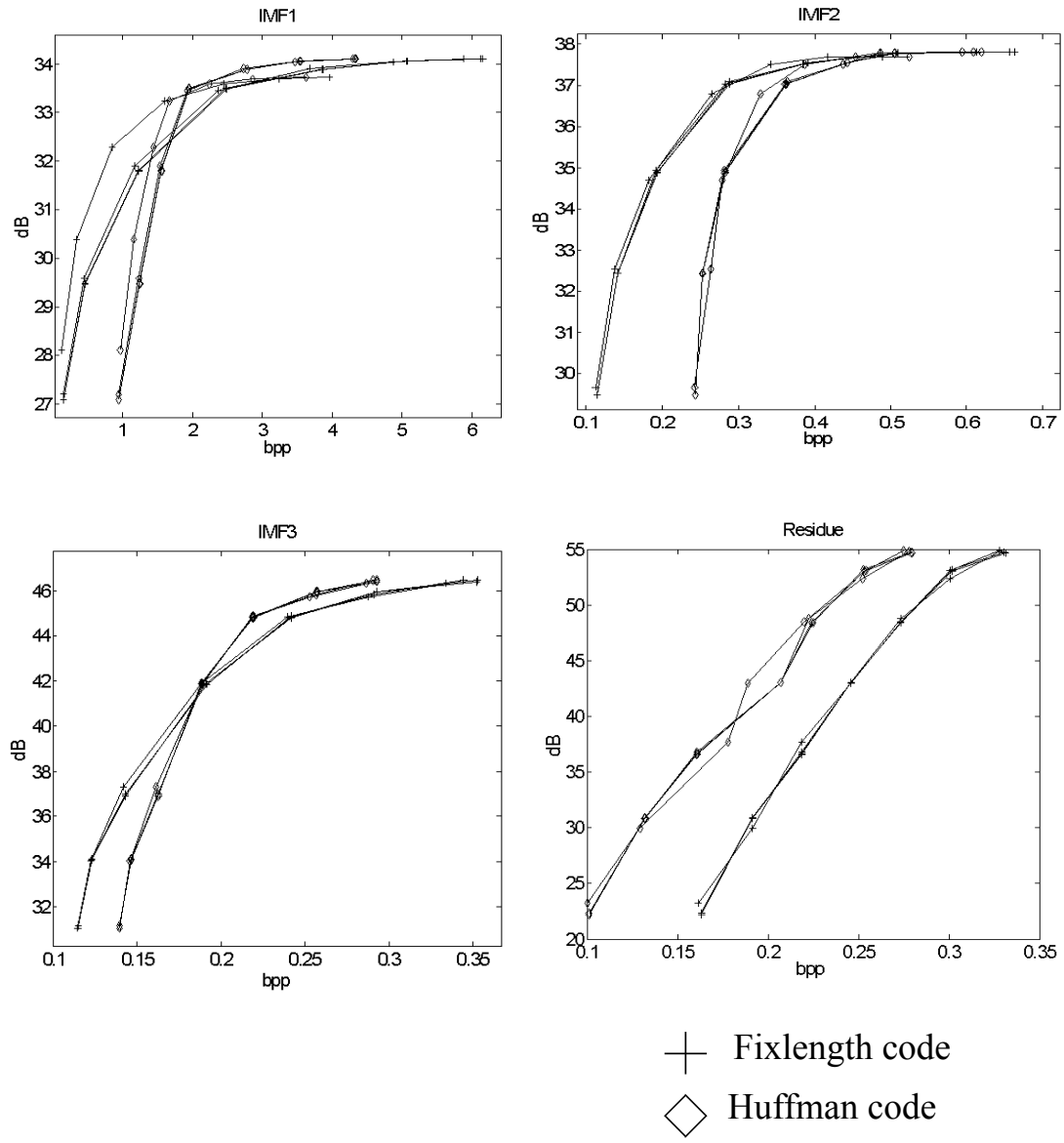


Figure 9.6. VSDCTEMD Coding result of Detail two. Zero zone=1, 10, 20 quantization varies from 256 levels down to 8 levels

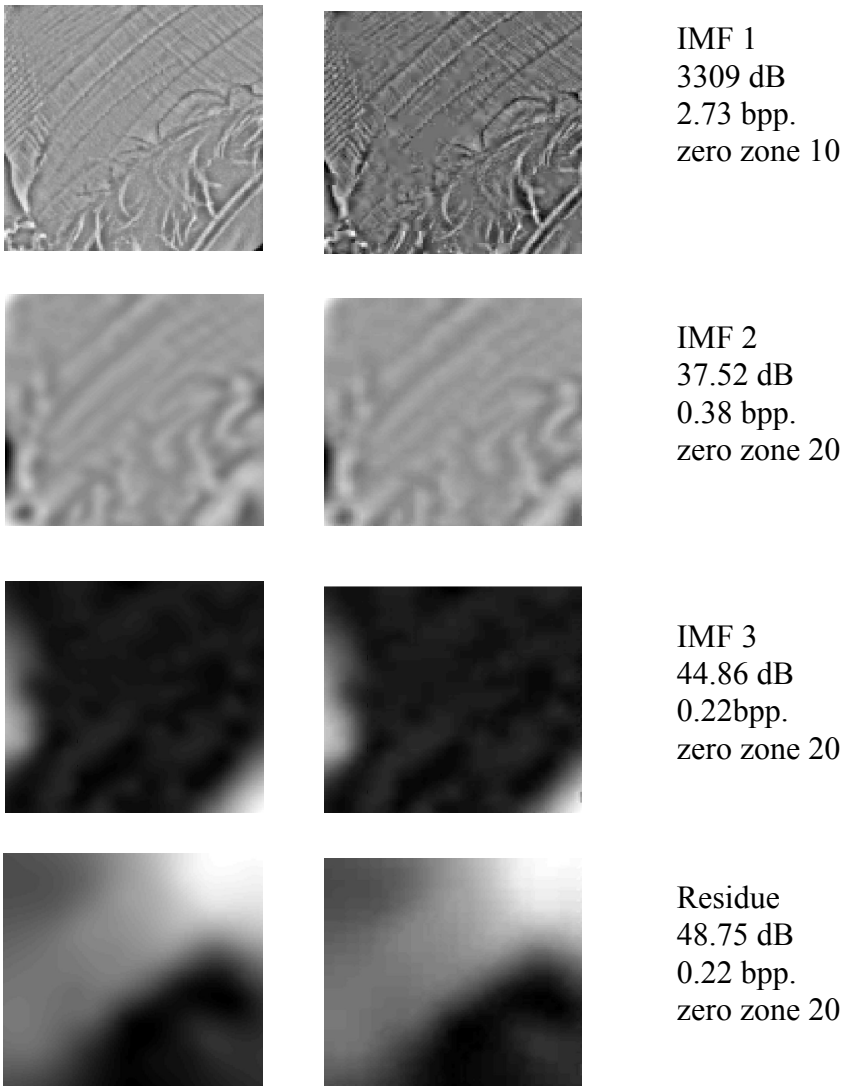


Figure 9.7. The result of VSDCTEMD coding of Detail two with different values on the zero zone. Left: original, Middle: reconstructed, Right: sampling rate and distortion.

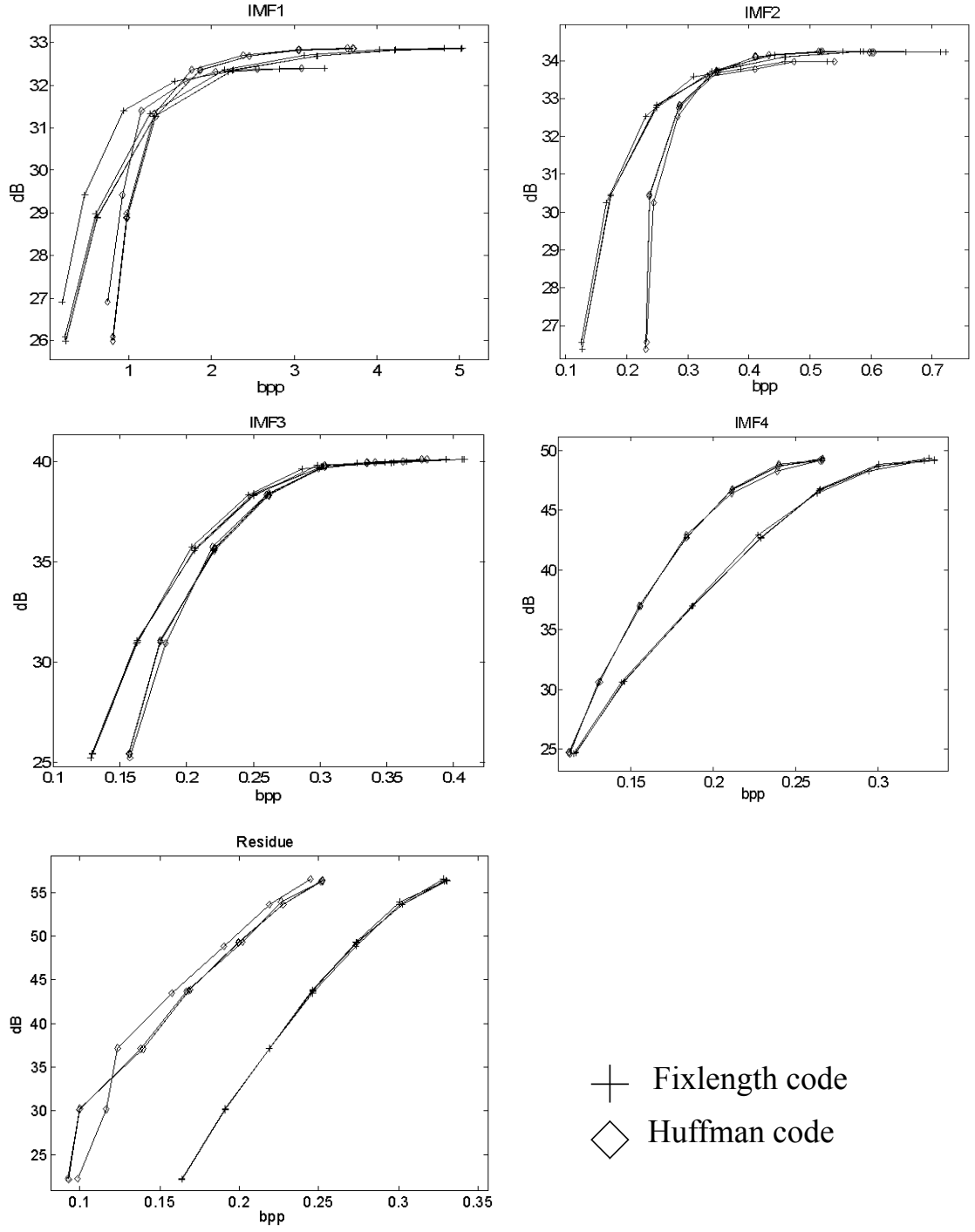


Figure 9.8. VSDCTEMD Coding result of Lenna128. Zero zone=1, 10, 20 quantization varies from 256 levels down to 8 levels.

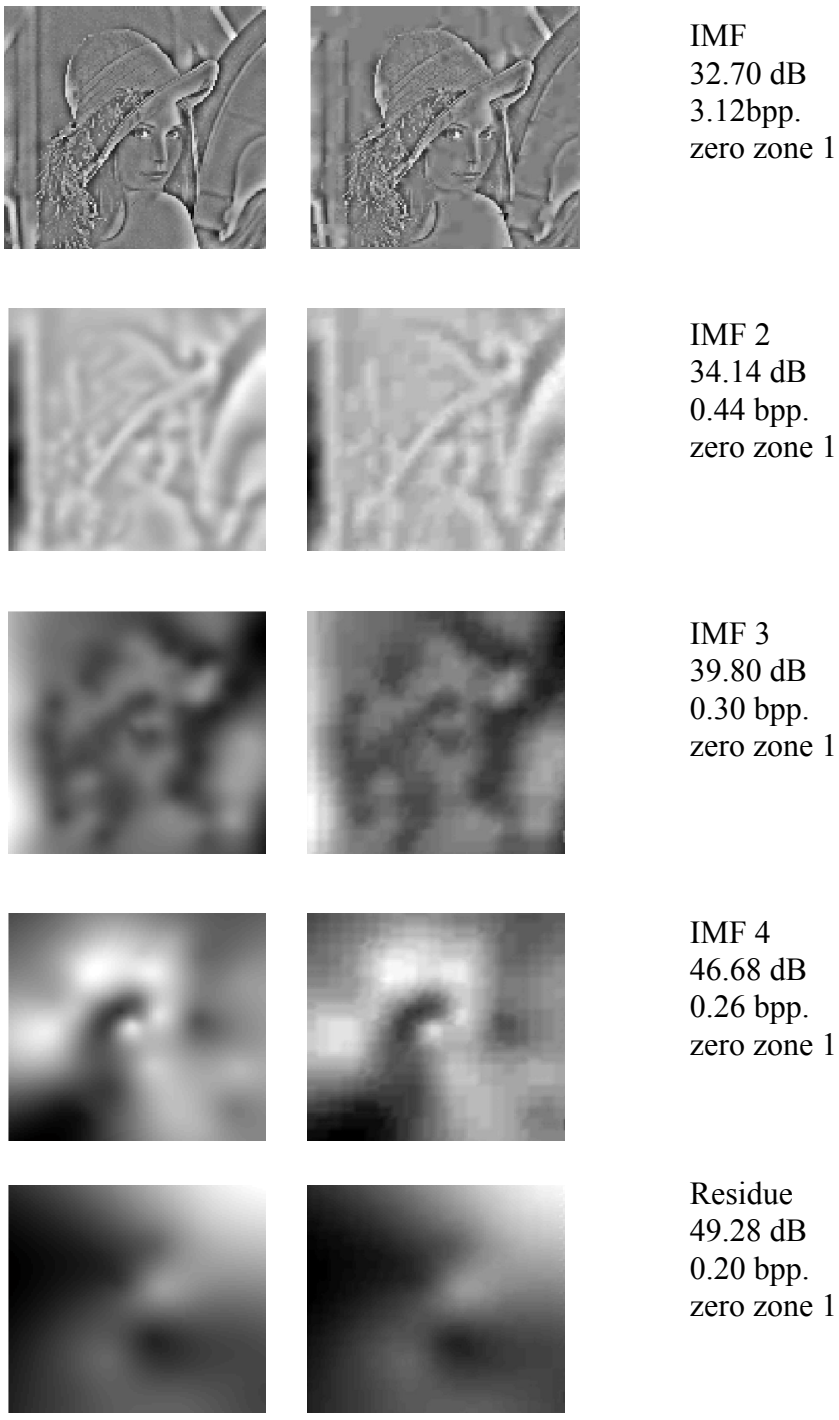
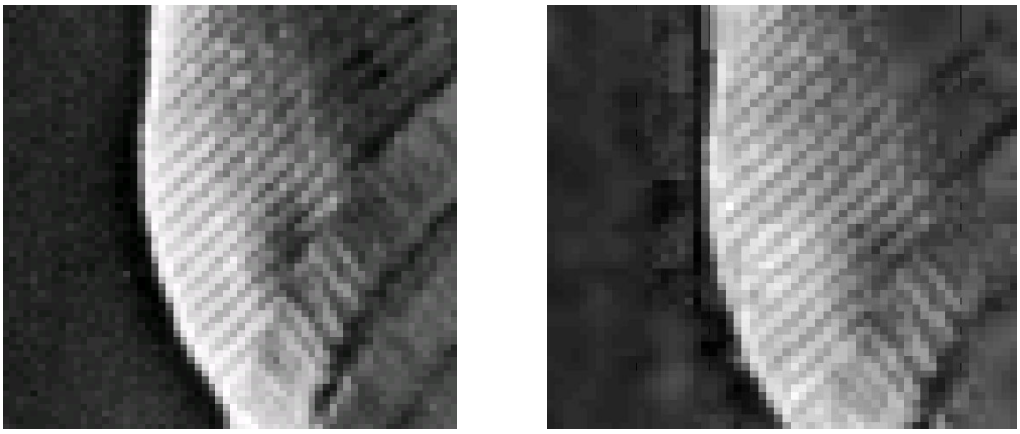
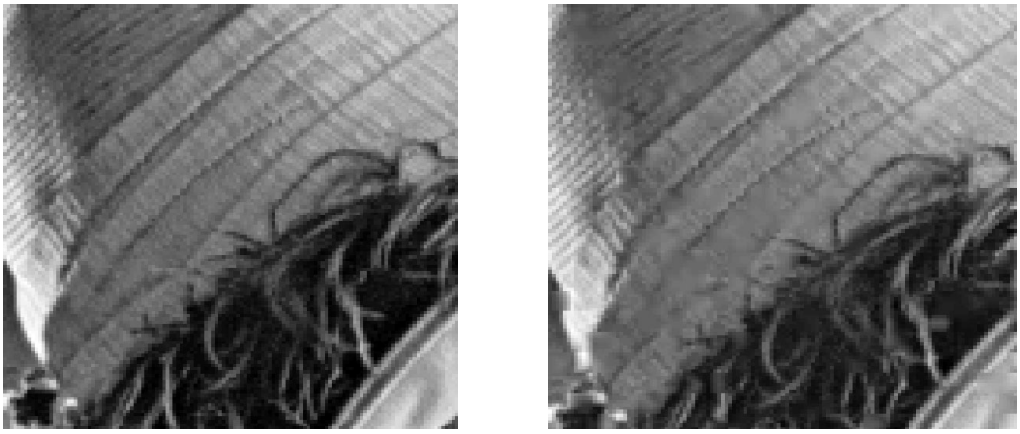


Figure 9.9. The result of VSDCTEMD coding of Lenna128. Left: original, Middle: reconstructed, Right: sampling rate and distortion.



a b

Figure 9.10. a) Detail one original, b) Detail one image reconstructed to 28.27 dB using 4.79 bpp.



a b

Figure 9.11. a) Detail two original, b). Detail two image reconstructed to 30.76 dB using 3.56 bpp.

a b

Figure 9.12. a) Lenna128 original, b) Lenna128 image reconstructed to 29.33 dB using 4.33 bpp.

9.2 Image coding using VSDCTEMD and DCT threshold coding

In this modified coder we will only decompose into one IMF and one residue. We will use the VSDCTEMD from section 9.1 on the first IMF and DCT threshold coding described in section 6.4 on the first residue.

9.2.1 Coder outline

This coder uses only the first IMF and the corresponding first residue to code the image. The coder is shown in Figure 9.13.

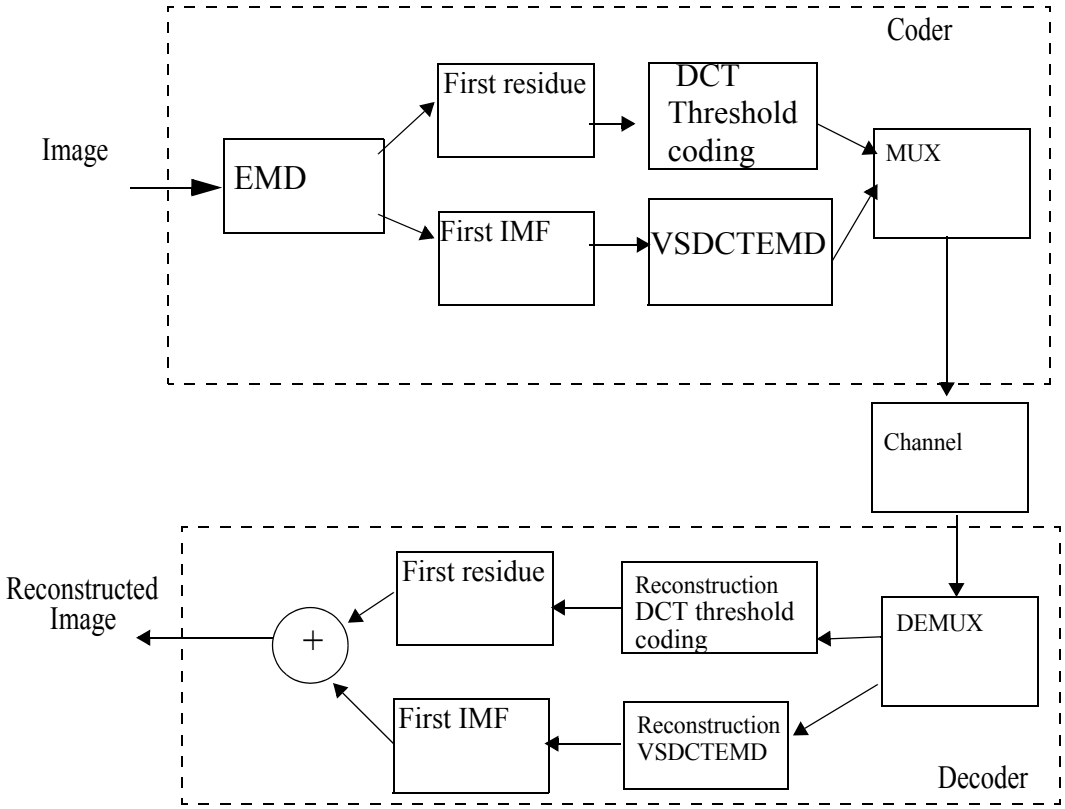


Figure 9.13. Combined VSDCTEMD and DCT threshold coding scheme.

The first IMF is coded with the VSDCTEMD. The details of this block are shown in Figure 9.2. The first residue is coded using DCT threshold coding. The details of this coder are shown in Figure 6.26. The reconstruction is done according to the reconstruction blocks of each coder, shown in Figure 9.3 and Figure 6.27 respectively.

9.2.2 Results

The coder is tested on the images Detail one, Detail two, and Lenna128. The result of the coding of the first IMF using VSDCTEMD and DCT threshold coding on the first residue is shown in Figure 9.14, Figure 9.16 and Figure 9.18 for the images respectively. All pos-

sible pairs of these two sets of results for each image are presented in Figure 9.15, Figure 9.17 and Figure 9.19 as the result for this coder. Two different reconstruction examples for each image are presented in Figure 9.20, Figure 9.21 and Figure 9.22.

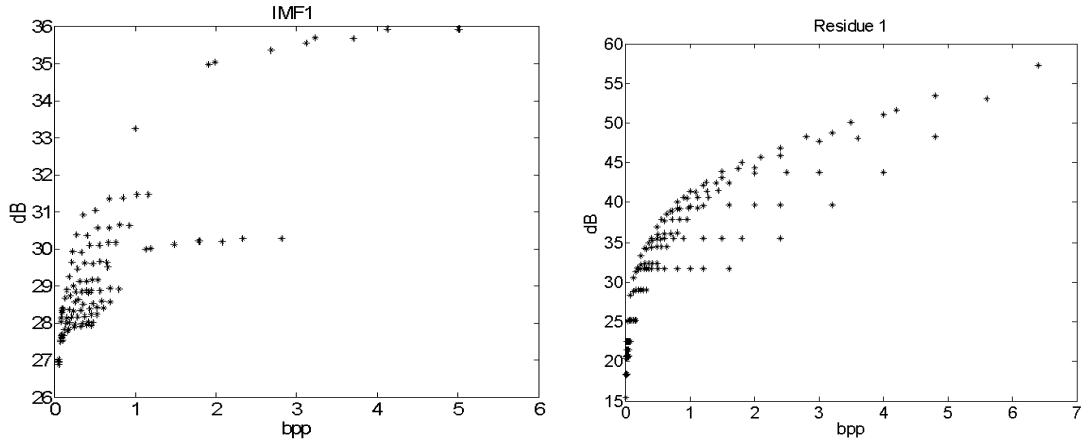


Figure 9.14. Result of coding Detail one by VSDCTEMD in first IMF and DCT threshold coding on first residue.

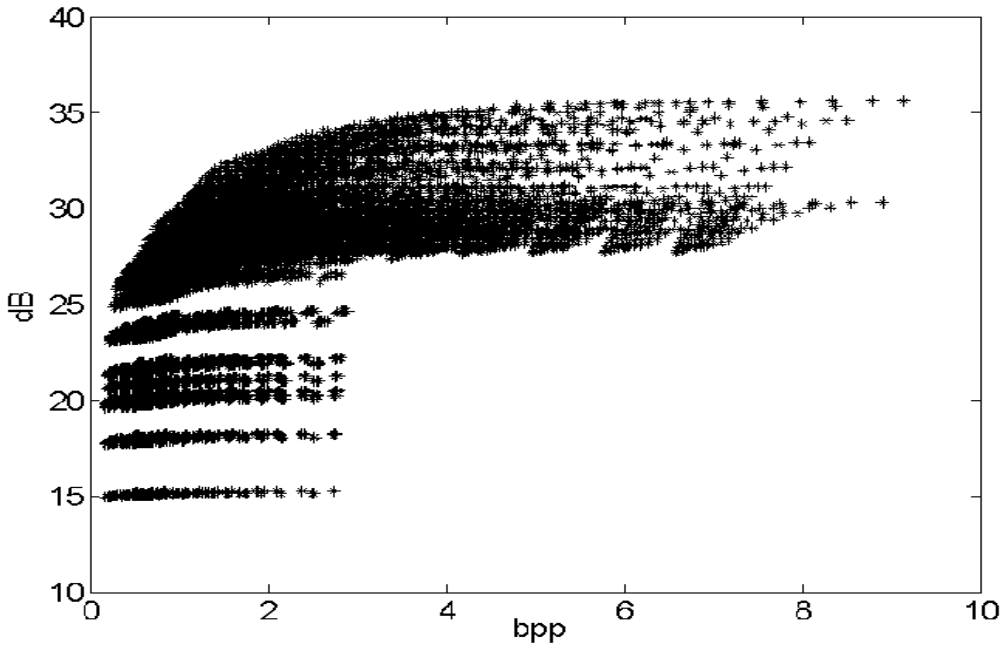


Figure 9.15. Result of combining the coding of Detail one by VSDCTEMD in first IMF and DCT threshold coding on first residue.

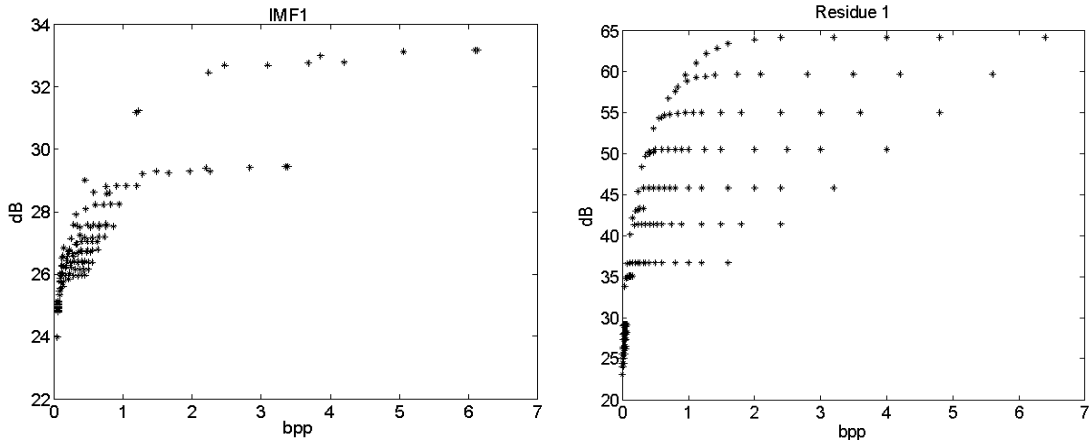


Figure 9.16. Result of coding Detail two by VSDCTEMD in first IMF and DCT threshold coding on first residue.

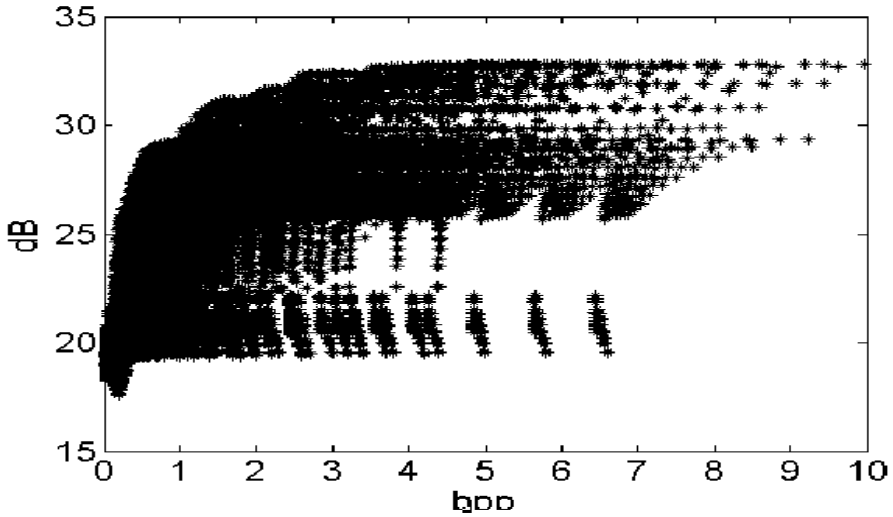


Figure 9.17. Result of combining the coding of Detail two by VSDCTEMD in first IMF and DCT threshold coding on first residue.

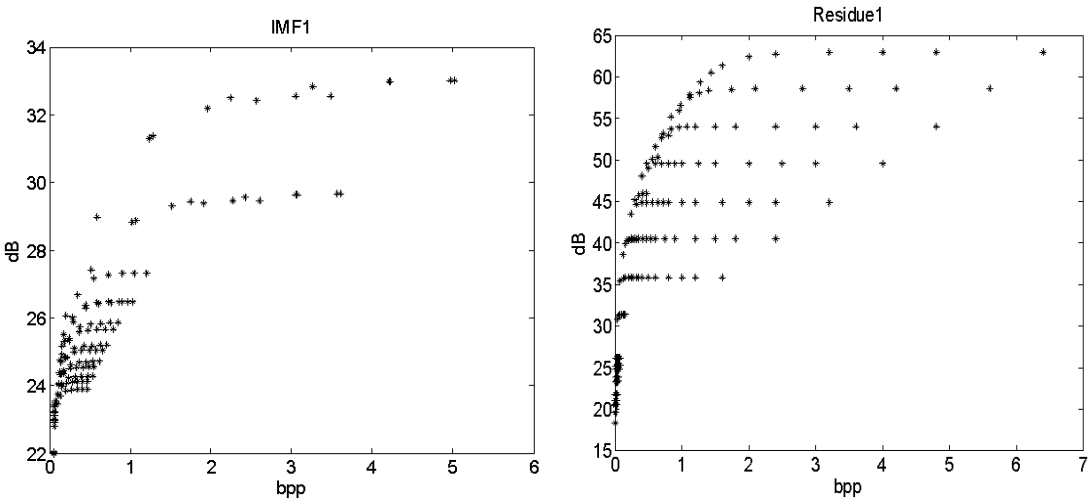


Figure 9.18. Result of coding Lenna128 by VSDCTEMD in first IMF and DCT threshold coding on first residue.

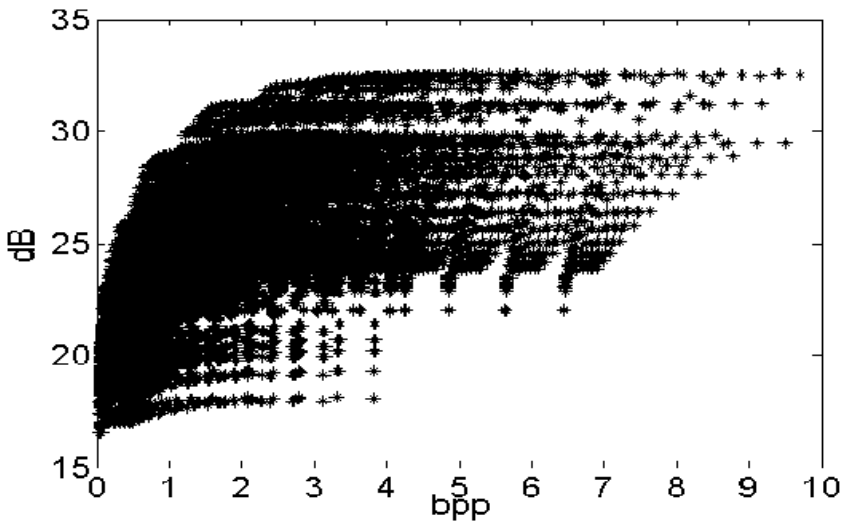
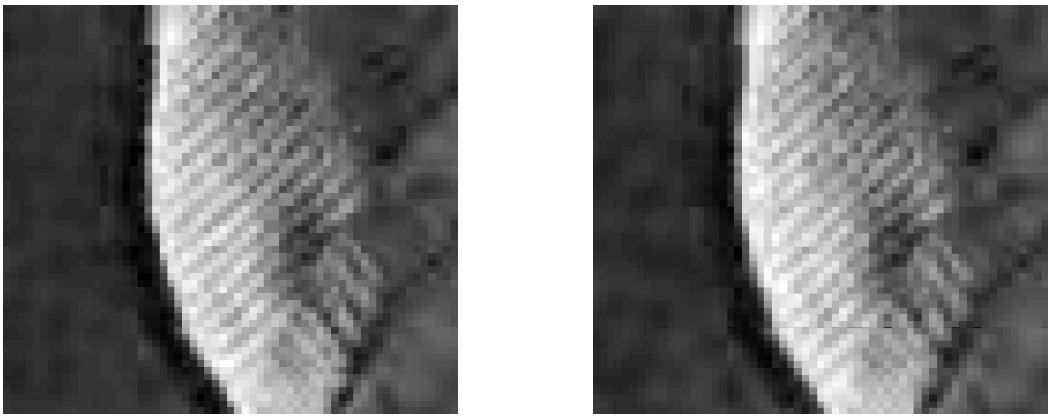


Figure 9.19. Result of coding Lenna128 by VSDCTEMD in first IMF and DCT threshold coding on first residue.



a b

Figure 9.20. Result of coding Detail one by VSDCTEMD in first IMF and DCT threshold coding on first residue.a) 1.00 bpp 30.09 dB, b) 0.79 bpp 29.23 dB

a b

Figure 9.21. Result of coding Detail two by VSDCTEMD in first IMF and DCT threshold coding on first residue.a) 1.2788 bpp 30.0208 dB, b) 0.69 bpp 28.88 dB

a b

Figure 9.22. Result of coding Lenna128 by VSDCTEMD in first IMF and DCT threshold coding on first residue.a) 1.35 bpp 30.52 dB, b) 0.89 bpp 28.86 dB

9.3 Summary

In this chapter we have used the variable sampling method for compression. We started with coding the samples of the IMF and residue with the DCT followed by an entropy coder. We then tried the approach of using only one IMF and coding the first IMF by using the DCT of the variable sampled blocks and coding the first residue using the threshold DCT method presented in Chapter 6.

Chapter 10

Empiquency-Controlled Image Coding

In this chapter we present alternative ways to code an image using the blockbased variable sampling. We assume that the same method that gives good results coding the first IMF also can be used on the image itself. We use the empiquency of the first IMF to control the choice of coding method and sampling rate used in the different blocks of the image. In the first scheme, we use the empiquency of the first IMF to control the variable sampling of the image itself. In our second approach we apply quantization and entropy coding on the variable sampling components. Finally in our third scheme we use the empiquency-controlled variable sampling together with DCT coding of the samples.

10.1 Empiquency-controlled variable sampling of the original image

This approach is based on the variable sampling technique described in Chapter 7. The difference is that the EMD process now stops when the first IMF has been found. The empiquency of the first IMF is used to control the subsampling of the image.

Coder outline

The coder is shown in Figure 10.1. The details of the VS block are presented in Figure 10.2. The first IMF is searched for extrema points and both the image itself and the maximum empiquency image is decomposed into overlapping 7x7 pixel blocks, as described in section 7.1.1. The value of the maximum empiquency found in the extrema point image block determines the sampling rate for the subsampling of the corresponding image block. The samples are quantized and Huffman coded. The details of the recon-

struction block are shown in Figure 10.3. The two-bit header indicates the sampling rate. With this information the components achieved from the stream can be placed in their proper place in the 7x7 pixel block. The stream only contain samples for the 6x6 block, the missing samples are found in the neighbouring blocks. The 7x7 blocks of samples are interpolated and the nonoverlapping 6x6 part of the reconstructed block is used when patching the blocks to reconstruct the image.

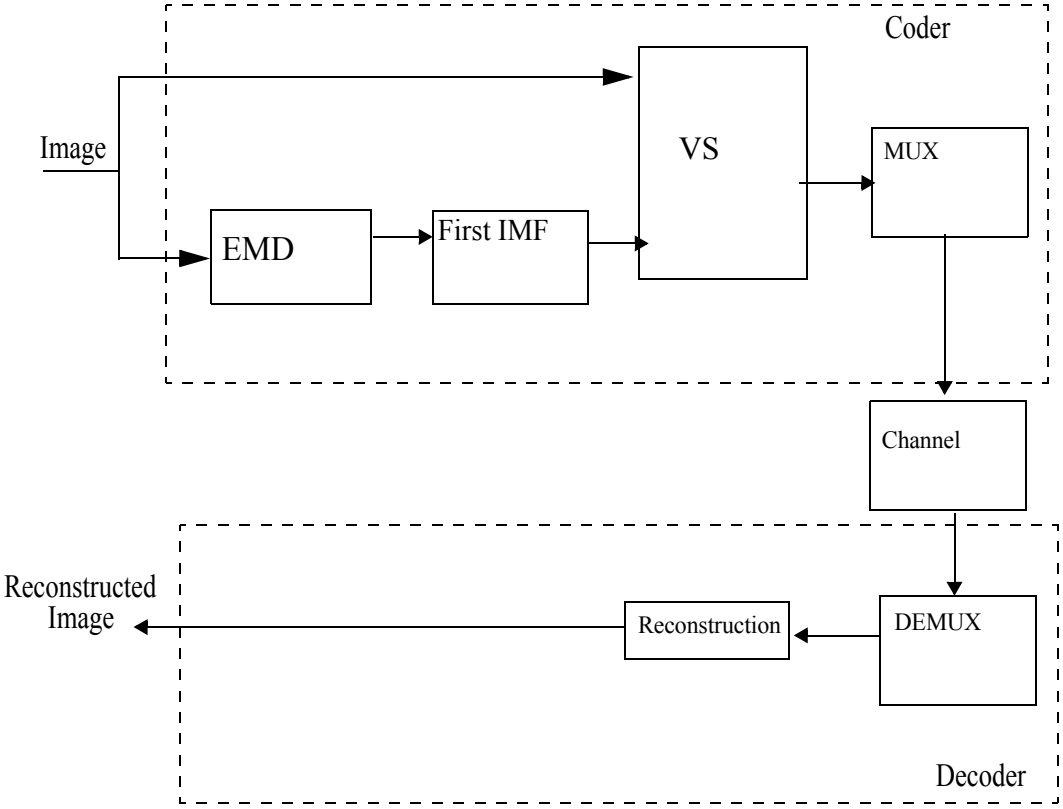


Figure 10.1. Empiquency controlled variable sampling scheme.

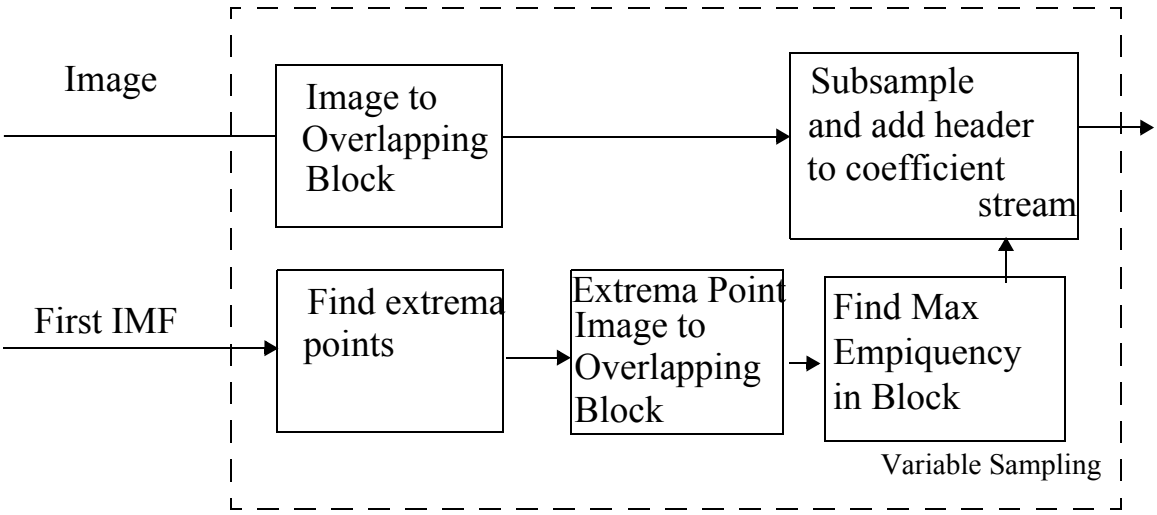


Figure 10.2. Details of the VS block.

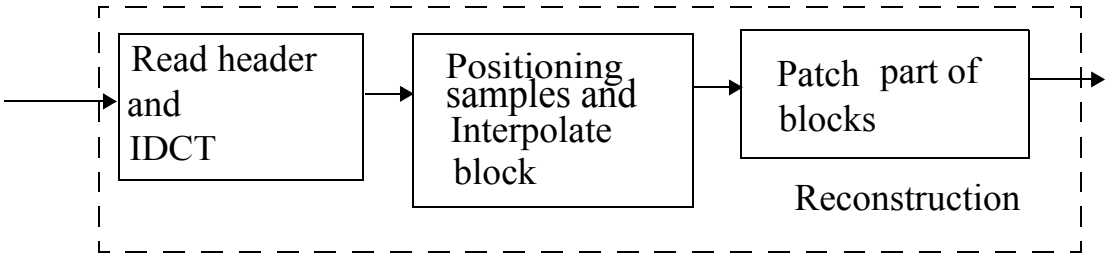


Figure 10.3. Detail of the reconstruction block.

10.1.1 Results

The empiquency-controlled variable subsampling of the image is tested on the images Detail one, Detail two, and Lenna128. The resulst of subsampling using zero zone 10 are shown in Figure 10.4, Figure 10.5 and Figure 10.6 for the images, respectively. Only the variable subsampling of the image renders compression to some degree. The image quality shown here is the best we can expect from the next two coders in this section as they only differ in that they apply compression algorithms to the samples.

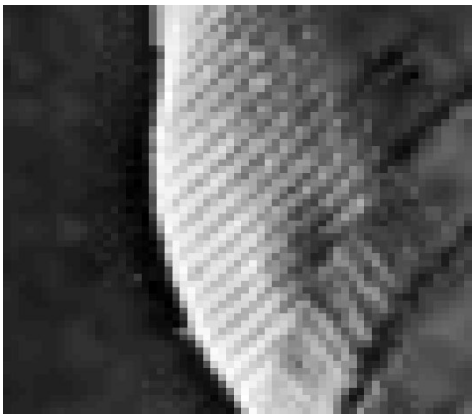


Figure 10.4. Variable sampling of Detail one, 34.60 dB at average sampling rate 0.71.

Figure 10.5. Variable sampling of Detail two, 32.93 dB at average sampling rate 0.86.

Figure 10.6. Variable sampling of Lenna128, 32.85 dB at average sampling rate 0.71.

10.2 Empiquency-controlled entropy coded variable sampling (Entropy coded VS)

The structure of the variable sampling in section 10.1 is inherited in this coding approach. The sampling process leaves us with a set of samples for each block which are quantized and entropy coded using Huffman code or fixlength code.

Coder outline

The coder is shown in Figure 10.7

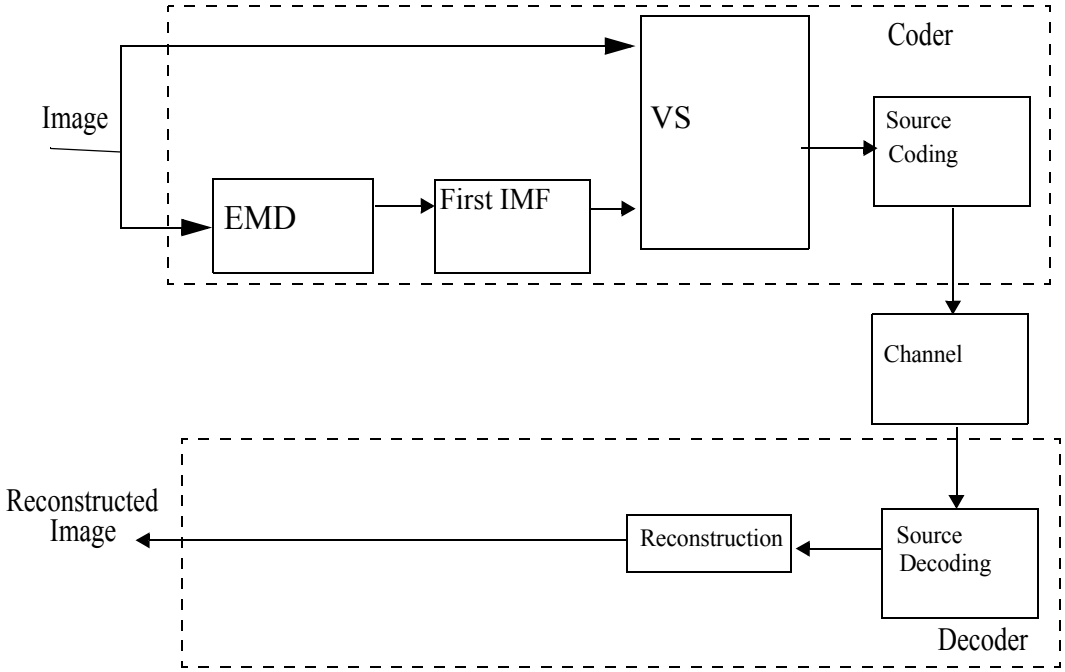


Figure 10.7. Coding scheme.

The first IMF is searched for extrema points and as in previous schemes using variable sampling, only the significant extrema points are kept for the definition of empiquency. The details of the VS block are shown in Figure 10.2. Both the image itself and the extrema point image are decomposed into overlapping 7x7pixel blocks as described in section 7.1.1. The value of the maximum empiquency in the extrema point image block decides the sampling rate for the subsampling of the corresponding image block. The samples are represented by one sample alone or 6x6, 3x3, or 2x2 blocks of samples. A two-bit header is added to the samples of each block to indicate the sampling rate of this specific block.

The details of the reconstruction block are shown in Figure 10.3. The two-bit header

indicates the sampling rate. With this information the components can be placed in their proper place in the 7x7 pixel block. The stream only contains samples for the 6x6 block, the missing samples are found in the reconstructed neighbouring blocks. The 7x7 blocks of samples are interpolated and the nonoverlapping 6x6 part of the reconstructed block is used to reconstruct the image.

10.2.1 Results

The coder is tested on three different images, Detail one, Detail two, and Lenna128, using zero zones 1, 10, and 20. The quantization varies from 256 levels to 4 levels. Figure 10.8, Figure 10.9, and Figure 10.10, show the coding result for the images respectively, and in Figure 10.11, Figure 10.12, and Figure 10.13, two example reconstructions for each image are presented.

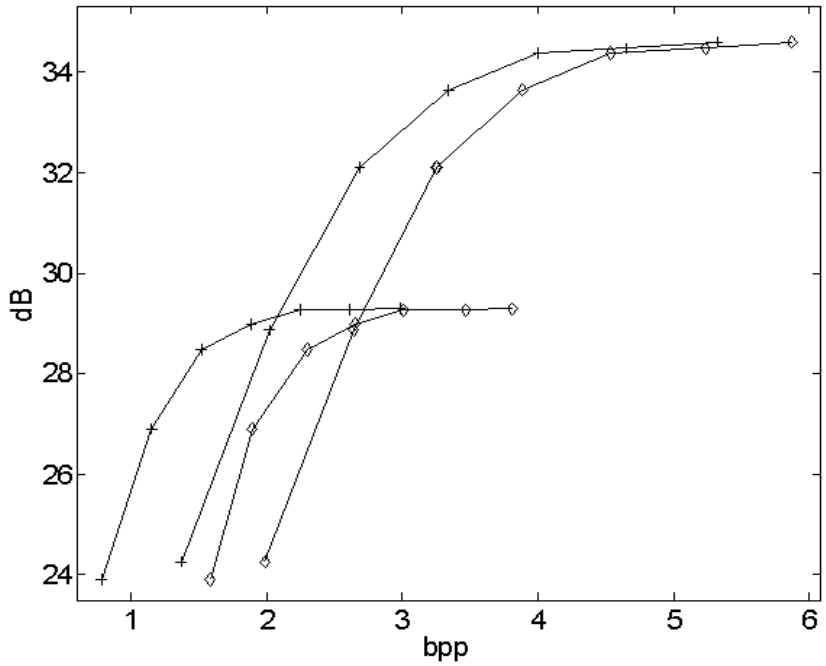


Figure 10.8. Entropy coding result of Detail one. Zero zone=1, 10, 20; quantization varies from 256 levels down to 4 levels

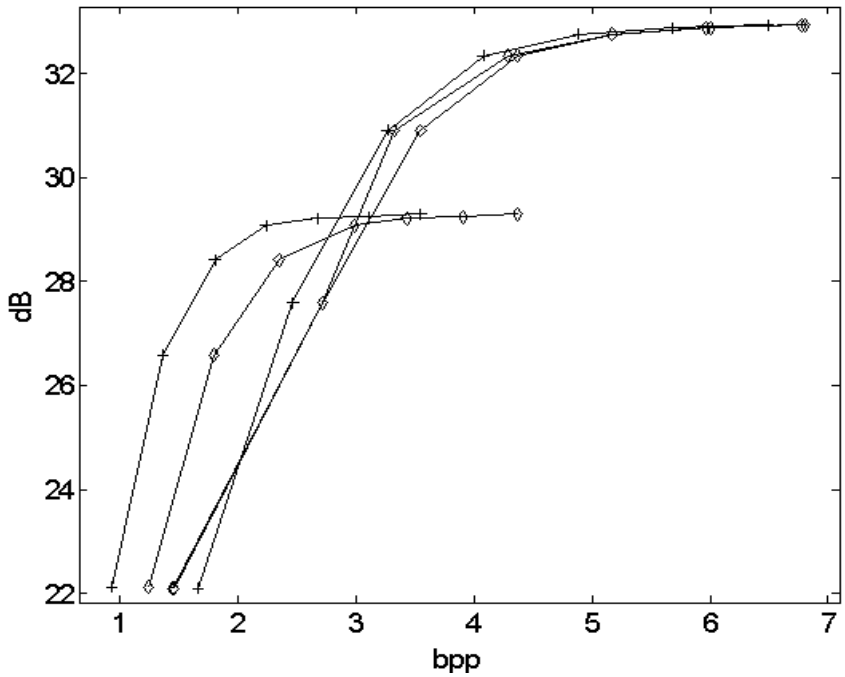


Figure 10.9. Entropy coding result of Detail two. Zero zone=1, 10, 20; quantization varies from 256 levels down to 4 levels

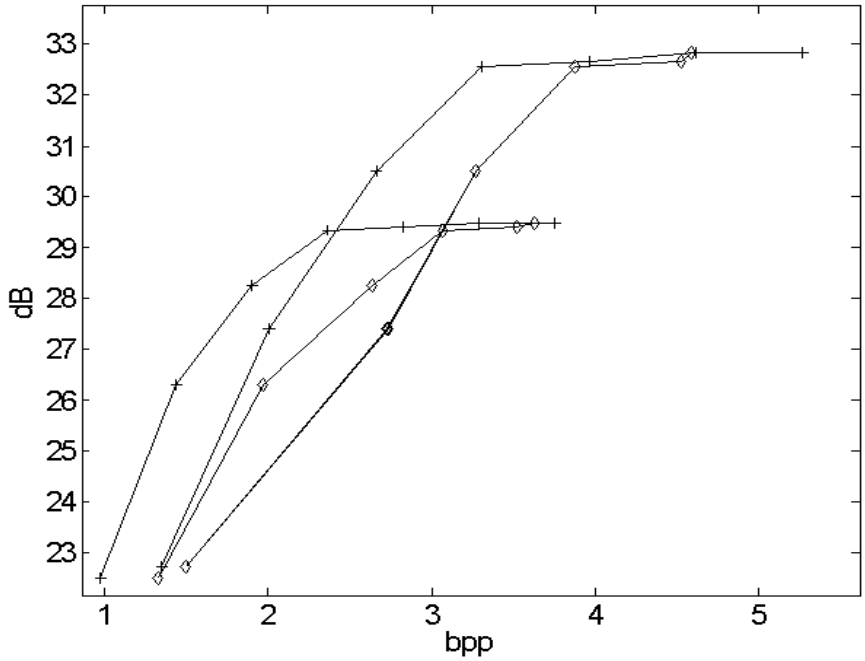
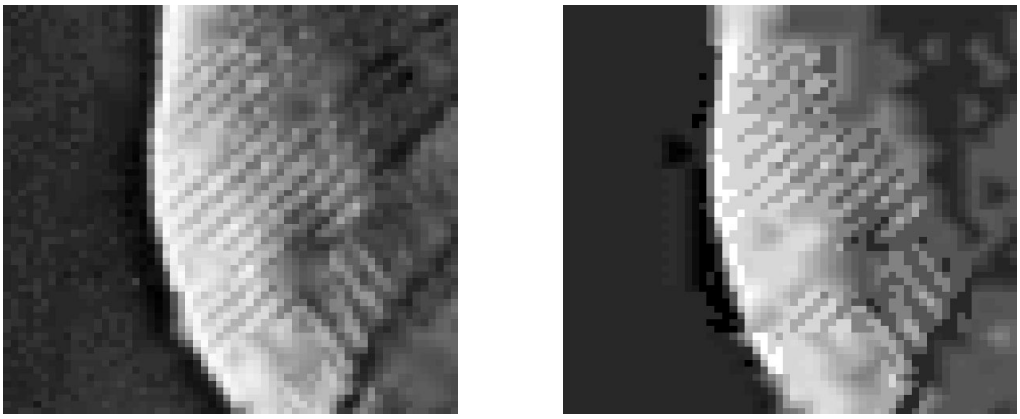


Figure 10.10. Entropy coding result of Lenna128. Zero zone=1, 10, 20; quantization varies from 256 levels down to 4 levels



a b

Figure 10.11. Entropy coded VS of Detail one: a) 4.00 bpp 34.38 dB, b) 1.15 bpp 26.90 dB

a b

Figure 10.12. Entropy coded VS of Detail two: a) 4.07 bpp 32.34 dB, b) 1.80 bpp 28.41 dB

a b

Figure 10.13. Entropy coded VS of Lenna128: a) 3.31 bpp 32.56 dB, b) 1.90 bpp 28.96dB.

10.3 Empiquency-controlled image coding using variable sampling and DCT coding (VSDCT)

The structure of the variable sampling in section 10.1 is inherited in this coding approach. The sampling process leaves us with a set of samples for each block which are squeezed into blocks of smaller size and DCT coded. The DCT components are then quantized and thresholded.

Coder outline

The VSDCT coder is shown in Figure 10.14

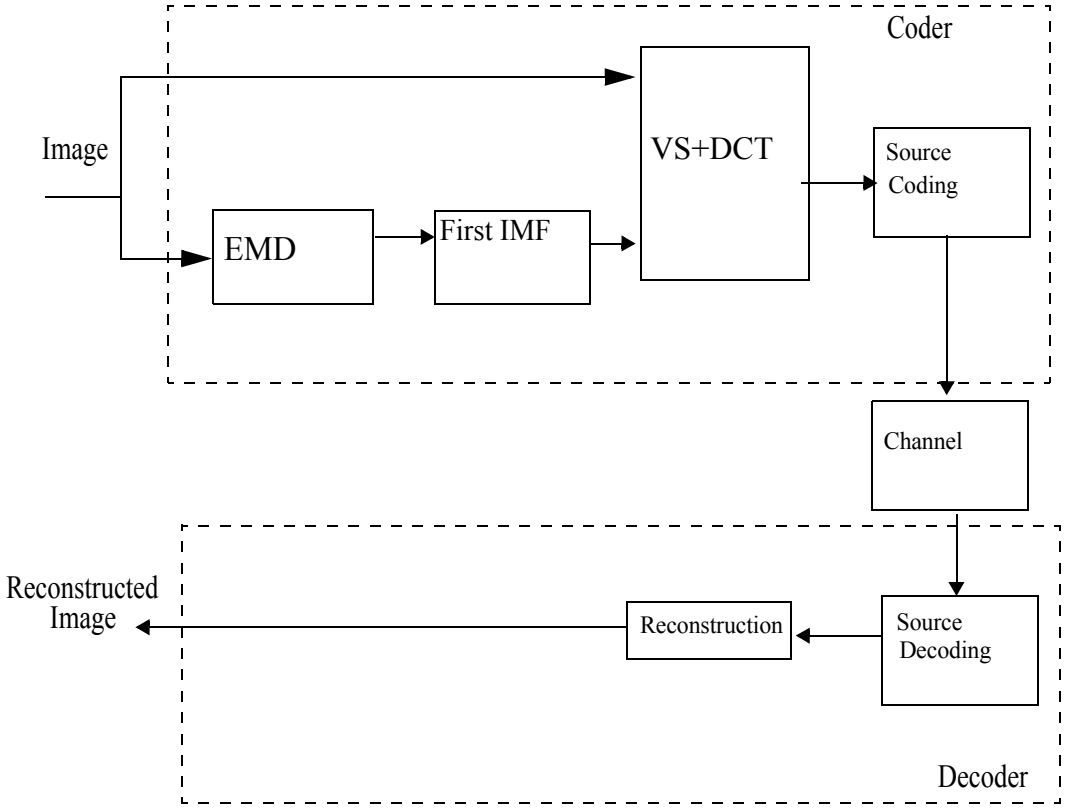


Figure 10.14. VSDCT coding scheme.

The first IMF is searched for extrema points and as in previous schemes using variable sampling, only the significant extrema points are kept for the definition of empiquency. The details of the VS+DCT block are shown in Figure 10.15. Both the image itself and the extrema point image are decomposed into overlapping 7x7 pixel blocks as described in section 7.1.1. The value of the maximum empiquency in the extrema point image block decides the sampling rate for the subsampling of the corresponding image block. The samples are represented by one sample alone or 6x6, 3x3, or 2x2 blocks of samples. These are DCT coded and the components are quantized and thresholded before the two-bit block header is added.

The details of the reconstruction block are shown in Figure 10.16. The two-bit header indicates the sampling rate. With this information the components achieved from the inverse DCT transform of the stream can be placed in their proper place in the 7x7 pixel block. The stream only contains samples for the 6x6 block, the missing samples are found in the reconstructed neighbouring blocks. The 7x7 blocks of samples are interpolated and the nonoverlapping 6x6 part of the reconstructed block is used to reconstruct the image.

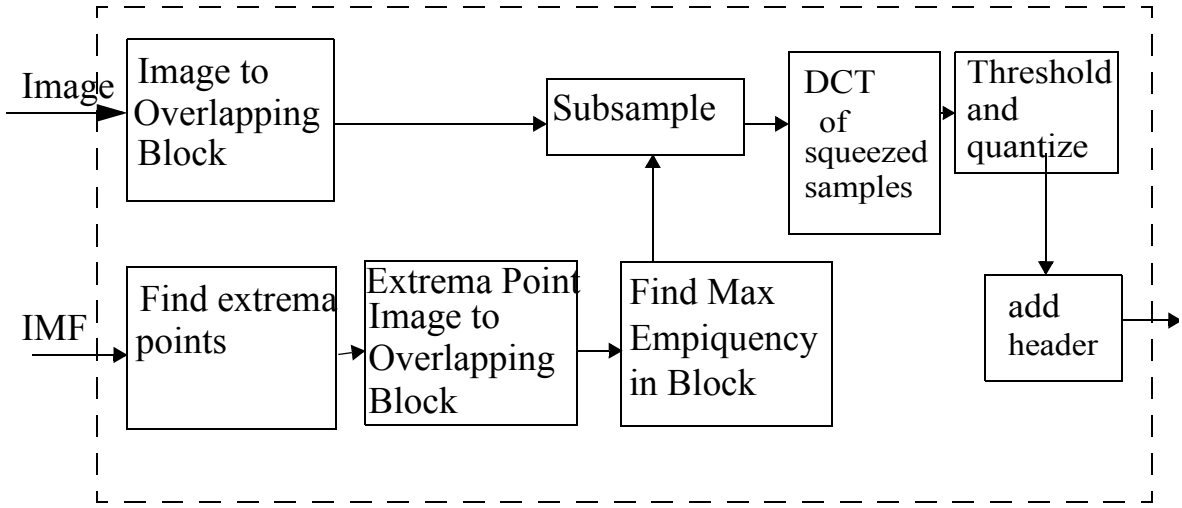


Figure 10.15. Detail of the VS+DCT coding block.

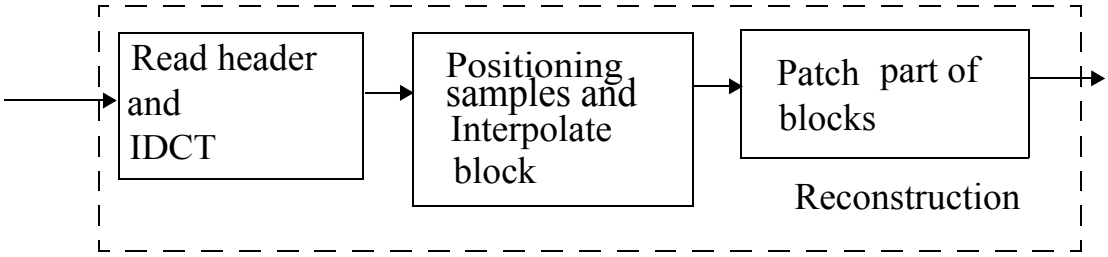


Figure 10.16. Details of the reconstruction block.

10.3.1 Results

The VSDCT coder is tested on three different images: Detail one, Detail two, and Lenna128, using zero zone 1, 10, and 20. The quantization varies from 256 levels to 8 levels. The DCT components are thresholded. The threshold varies from 0.1% to 100% of the maximum value. Figure 10.17, Figure 10.18, and Figure 10.19, show the coding result for the images respectively, and in Figure 10.20, Figure 10.21, and Figure 10.22, two example reconstructions for each image are presented. We see that this coder gives good compression results for bitrates around 1 bpp.

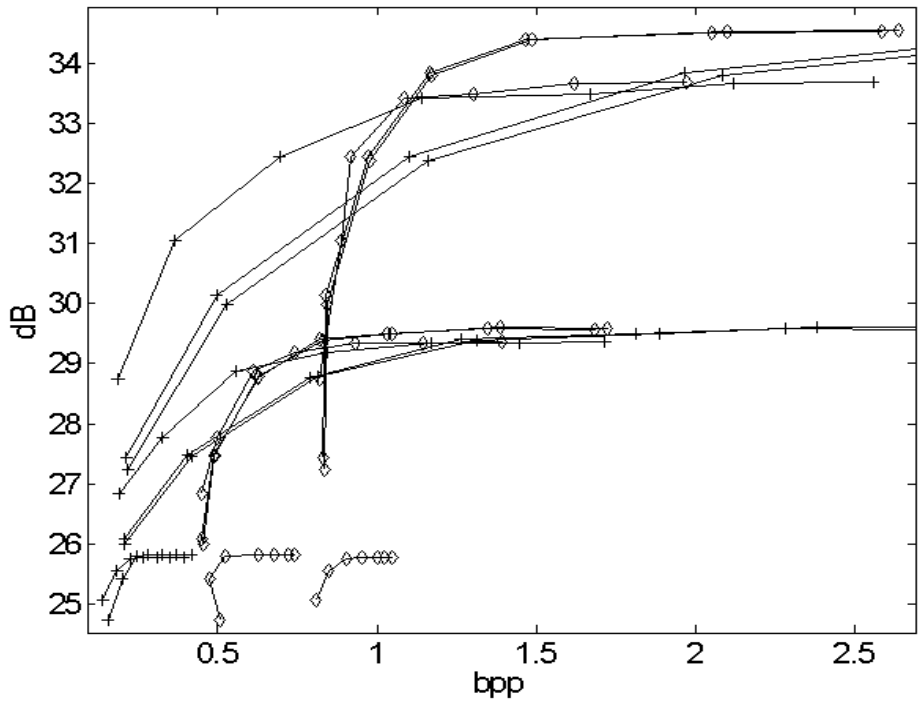


Figure 10.17. The result of using VSDCT coder on the Detail one image.

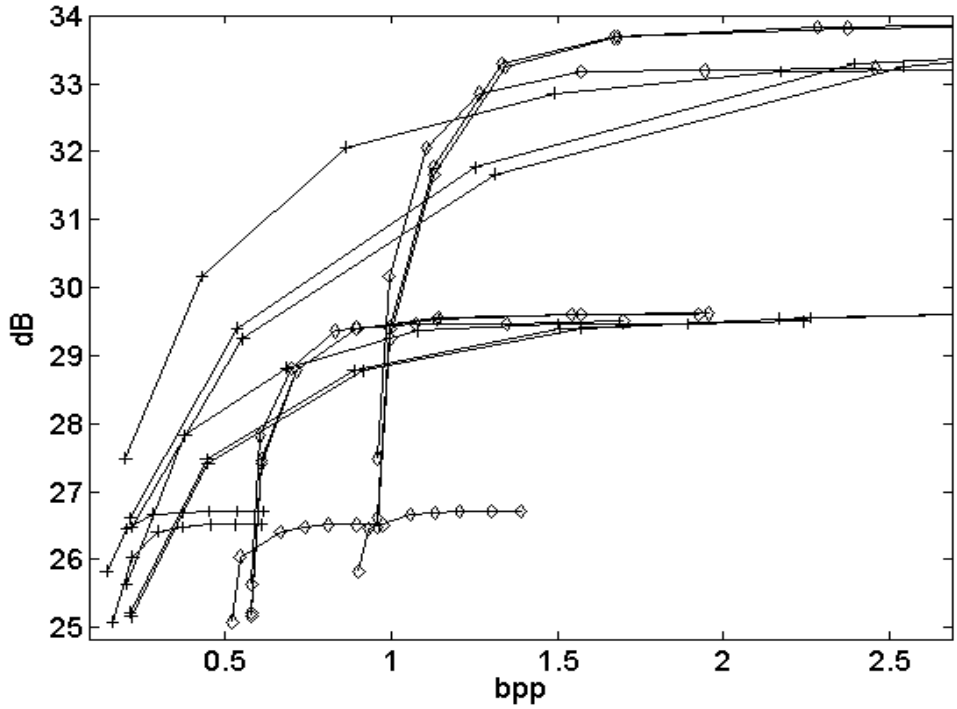


Figure 10.18. The result of using the VSDCT coder on the Detail two image.

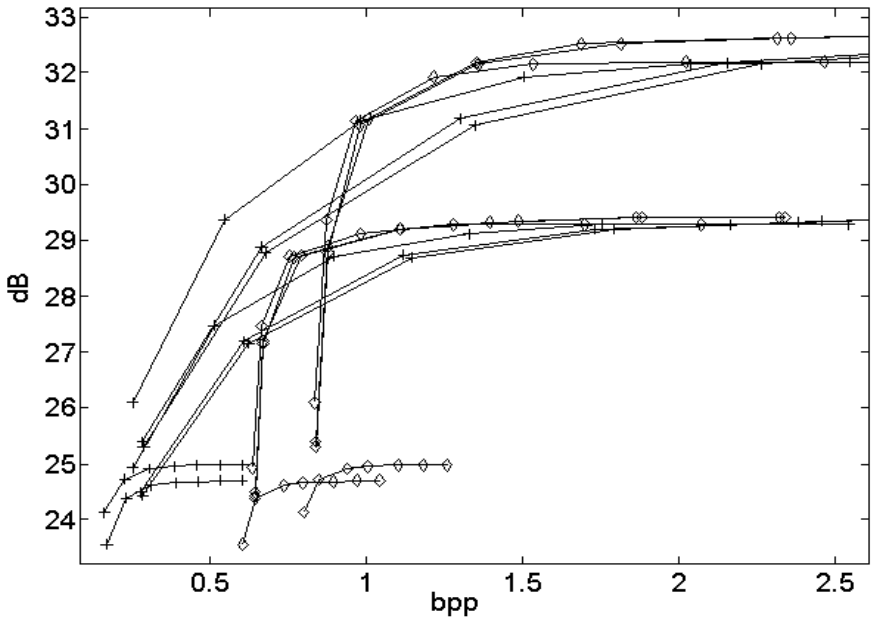
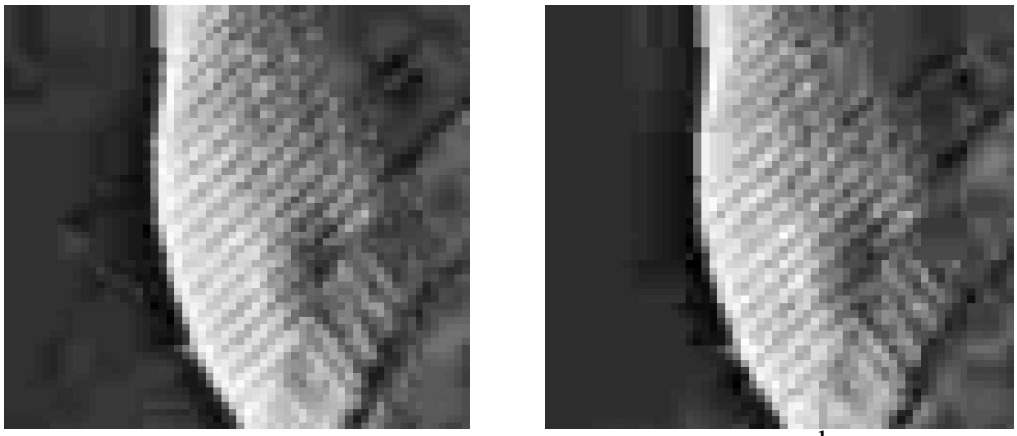
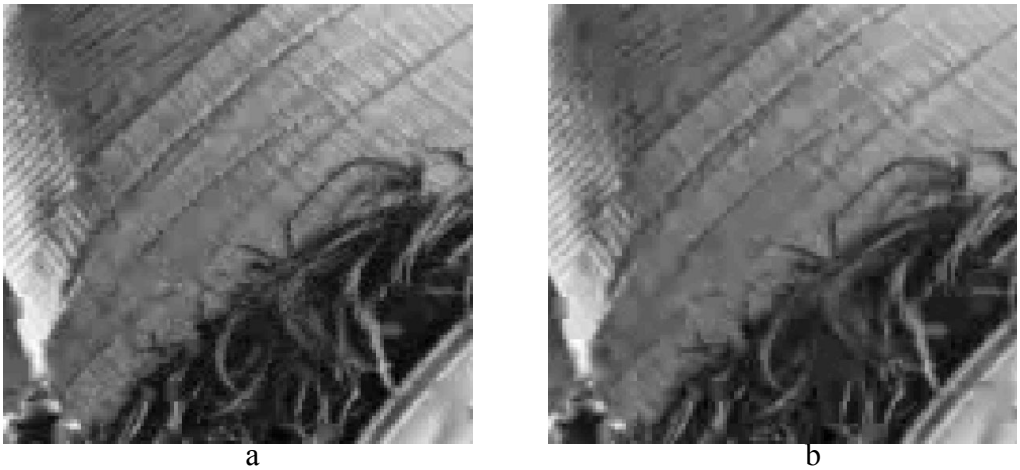


Figure 10.19. The result of using the VSDCT coder on the Lenna128 image.



a b

Figure 10.20. VSDCT of Detail one: a) 1.17 bpp 33.83 dB, b) 0.37 bpp 31.04 dB



a b

Figure 10.21. VSDCT of Detail two: a) 1.33 bpp 33.28 dB, b) 0.43 bpp 30.17 dB

a b

Figure 10.22. VSDCT of Lenna128: a) 1.35 bpp 28.93 dB, b) 0.66 bpp 21.17 dB.

10.4 Summary

In this chapter we have presented new methods to code an image using the blockbased variable sampling. The EMD is only used to find the first IMF. The empiquency of the first IMF is used to control the choice of coding method and sampling rate used in the different image blocks. First the empiquency of the first IMF was used to control the variable sampling of the image itself. This gives us the limit of the performance we can expect from the other coders tested. The variable subsampling of the image render compression to some degree. This is the basis for using the empiquency-controlled variable sampling together with two different entropy coders and the empiquency-controlled variable sampling together with DCT coding of the samples. The key results from each coder are summarized in Figure 10.23, Figure 10.24, and Figure 10.25.

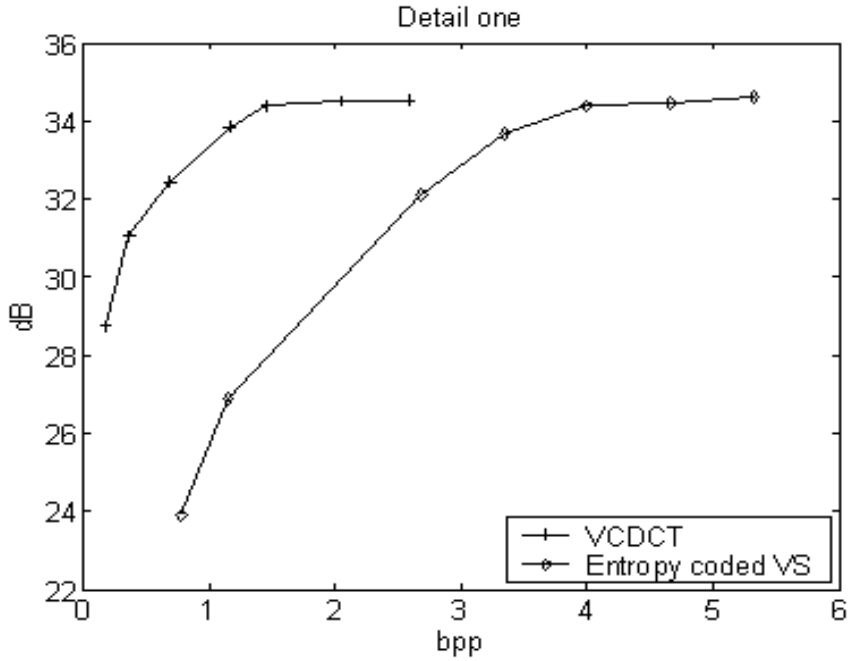


Figure 10.23. Summary of results from the different coders applied to the Detail one image.

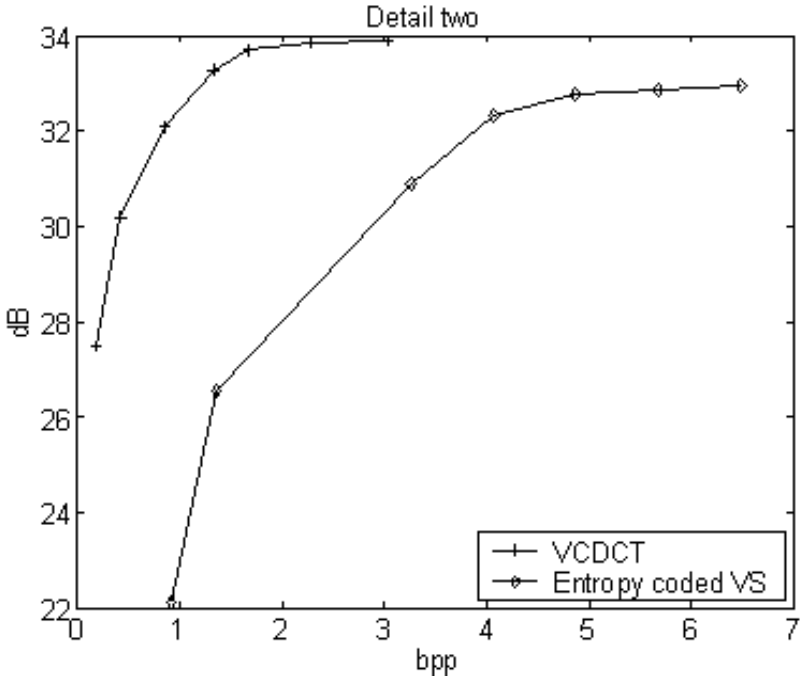


Figure 10.24. Summary of results from the different coders applied to the Detail two image.

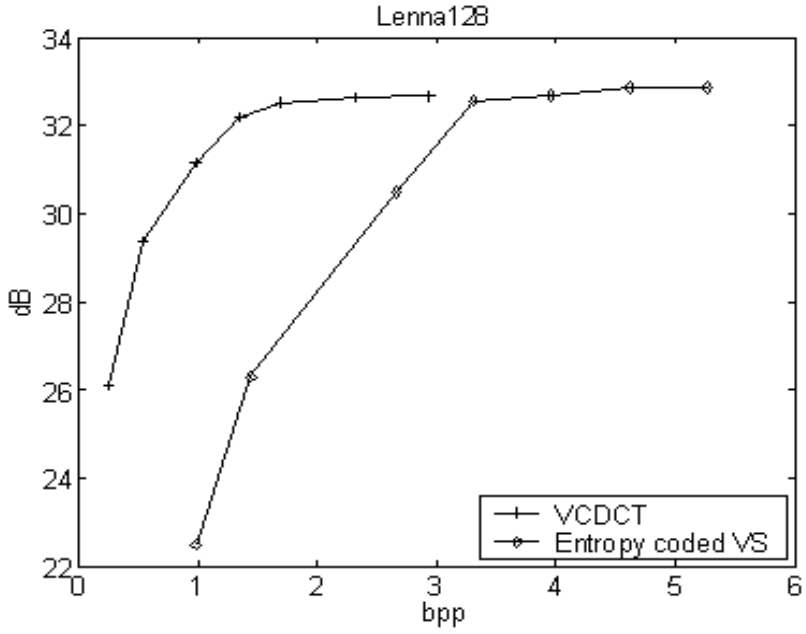


Figure 10.25. Summary of results from the different coders applied to the Lenna128 image.

Chapter 11.

Summary and Discussion

In this chapter we summarize the work in the second part of this thesis, trying to put it in a wider perspective, discussing the results and relating the work to the work of others.

11.1 EMD

The second part of this thesis discusses the Empirical Mode Decomposition (EMD), which is an adaptive decomposition with which any complicated signal can be decomposed into its Intrinsic Mode Functions (IMF), corresponding to the physical reality that generated the signal [36]. The purpose of the work on EMD on time signals in this thesis is to introduce the EMD concept.

An IMF is characterized by some specific properties. One is that the number of zero crossings and the number of extrema points are equal or differ only by one. Another property of the IMF is that the envelopes defined by the local maxima and minima, respectively, are locally symmetric around the envelope mean. The original purpose for the EMD was to find a decomposition which made it possible to use the instantaneous frequency, defined as the derivative of the phase of an analytic signal, in the time frequency analysis of the signal. Traditional Fourier analysis does not allow a signal's spectrum to change over time. The EMD, however, provides a decomposition method that analyses the signal locally. The use of EMD in the Hilbert-Huang Transform (HHT) [36] is viewed as an example of EMD for time signals. Different interpolation methods are discussed for time signals as an introduction to the same problem for images.

Previous work often mention the lack of a mathematical formalism to describe the EMD, the concept is truly empirical. In this thesis we keep the empirical approach as we extend the EMD concept into two dimensions.

The sifting process of [36] is here extended and given a more strict formulation in two dimensions.

The behaviour of the EMD as a filter bank is highlighted in Flandrin et al. [26] through the analysis of noise. The first IMF resemble a “high-pass filtered” noise signal. The rest

of the IMFs resembles “band-bass filtered” noise signals with their centre frequency decreasing in an octave band manner, just like a filter bank. The experiment shows that the EMD sorts the energy of the signal similarly to a subband coder. The same can be seen when EMD is applied to images. The tendency is that the IMFs other than the first are low frequency images. By examining of the DCT of the whole image we see that for the first IMF the energy is spread all over the matrix, with a highlight in the centre part representing the dominating spatial frequencies. For the subsequent IMFs and the residue the energy concentrates in the upper left corner, where it represents low frequency.

Some implementations of the EMD in two dimensions generate a residue with many extrema points [51,62,63]. These methods have not proven capable of fully decomposing the two-dimensional signal down to a residue with a low number of extrema points. In this thesis we present an improved method that can decompose the image into a number of IMFs and a residue with none, or with only a few extrema points. This method makes it possible to use the EMD for image processing.

For two-dimensional signals there are several possibilities to define extrema, each one yielding a different decomposition. In this thesis the extrema points are simply extracted by comparing the candidate data point with its nearest 8-connected neighbours. Clearly, more sophisticated methods could be used, but the extrema points defined by an 8-connected neighbourhood serve the purpose for EMD at this stage, with further improvement being possible. In Nunes et al. [63] the extrema points are selected using morphological reconstruction. Neither of these methods allow for saddle points to be considered as extrema points. Nunes et al. proposed the use of a watershed operation to find these extrema points in the future [63]. In this thesis we note that in the case where saddle points are considered to be extrema points, these are both maxima and minima at the same point. This nails the saddle point to the zero mean level in the first sifting round. For this reason, saddle points are not considered to be extrema points. Future work may find methods for extrema point extraction that could also handle saddle points.

In the second step in the sifting process the problem is to fit a surface to the two-dimensional scattered data points representing the extrema points. The interpolated surface must go through each data point. Overshot shall be avoided and the second derivative must be continuous everywhere for the signal to be smooth enough for two-dimensional EMD. The interpolation methods considered here are triangle-based cubic spline interpolation and thin-plate smoothing spline interpolation. The first is an example of an interpolation method using a piecewise approach to interpolate the surface. It produces piecewise smooth surfaces and is based on Delaunay triangulation [67] of the data. The piecewise approach causes more problems in the two-dimensional case than with one-dimensional signals. Even though the interpolators have continuous second derivatives, such as the cubic spline in [51] or the radial basis functions used in [63], the borders of the neighbouring pieces cause problems. It is possible that the piece-wise approach will work if we use another set of extrema points where the saddle points are included. This is not examined in this thesis but is left to future work.

The border constraints are even more important in two dimensions than they are in the one-dimensional EMD case and are the main objection to using spline interpolation in the EMD for two-dimensional signals. The set of extrema points is very sparse and since the interpolation methods only interpolate between these points, the borders need special care. In this thesis we introduce the trick of adding extra data points at the borders to the set of extrema points. These extra points are placed at the corners of the image and equally spaced around the border. Without these extra points, the areas not covered by

the interpolation traverse into the image in the sifting process.

In this thesis the thin-plate smoothing spline interpolation [8] is suggested for two-dimensional EMD analysis. This algorithm calculates the surface over the entire image and gives a surface with continuous second derivatives everywhere. The determination of the smoothing spline involves the solution of a linear system with as many unknowns as there are data points. This method turns out to successfully decompose an image into its IMFs and a smooth residue with no or only a few extrema points.

In [36] the sifting process stops when the difference between two consecutive siftings is smaller than a selected threshold. In this thesis the process stops when the envelope mean signal is close enough to zero as suggested in Linderhed [51]. The reason for this choice is that forcing the envelope mean to zero will guarantee the symmetry of the envelope and the correct relation between the number of zero crossings and number of extremes that define the IMF. A slightly more complicated version of this stop criterion is presented in Rilling et al. [73] along with a discussion of typical values on the threshold. The number of IMFs achieved with the EMD depends on the choice of value of the stop criterion. If it is too large, the number of IMFs is lower. If it is too small, the number of IMFs is large enough but computation time is longer than necessary.

11.2 Empiquency

The EMD is a truly empirical method, not based on the Fourier frequency approach but related to the locations of extrema points and zero-crossings. Using the EMD for signal analysis requires a mind free from the traditional Fourier-based frequency concept. Based on this we introduce the concept of *empiquency* [53], short for empirical mode frequency, instead of a traditional Fourier-based frequency measure to describe the signal oscillations. The measure of empiquency is defined as "One half the reciprocal distance between two consecutive extrema points". With this definition the values relate to normalized frequency used with discrete Fourier transform. We define d to be the distance between two neighbouring extrema points. In [36] the concept of *time scale* is used to describe the signal oscillations. In relation to empiquency this can be seen as the mean of all d in the signal. In this context the empiquency is the *local time scale.*

The IMF have the very special property of being locally zero mean and ideally having not more than one extrema point between two neighbouring zero-crossings. Because of this relationship between zero-crossings and extrema points there is also a relation between the empiquency, time scale, and the concept of *sequency* [31]. Sequency is defined as, "the average number of zero crossings per second divided by 2". For a sine or cosine the frequency and the sequency are the same, but the concept of sequency has a meaning for other signals such as Walsh functions or IMF as well. There is a possibility that the mathematical theory developed for sequency could be used for the development of mathematical methods for EMD. This idea is not further treated in this thesis but left to future work.

The empiquency value is assigned to every position between the two extrema points used to compute it. Each extrema point influences more than one set of empiquency definitions. We let it take the highest value of the empiquency values it defines. The empiquency is accompanied by an amplitude describing the signal at the actual position. At each extrema point the empiquency amplitude A_e is the absolute value of the extrema

point. At any other position the empiquency amplitude takes the value of the absolute mean of the extrema points that define the empiquency for the actual position. The empiquency concept will be further developed towards a texture analysis tool. In the image coding applications in this thesis the empiquency is only used as an indication of the highest spatial frequency present in the block to be coded.

In this thesis we have also discussed the selection of *significant extrema points*. This has turned out to be essential for image compression with EMD similar to the definition of significant coefficients in zerotree coding [75]. The selection of significant extrema points can also be used as a tool for noise reduction.

11.3 Variable sampling

It is known [59,35] that a band-limited signal can be uniquely determined from its non-uniform samples, provided that the average sampling rate exceeds the Nyquist rate. From this we assume that the IMF can be subsampled with special requirements on the sampling rate and that this rate is given by the maximum empiquency.

The special property of the IMF, namely that the empiquency varies, is used for variable sampling. The IMFs are smoother than the image itself; only the first IMF holds the nonsmooth parts of the image. This means that it should be possible to subsample the IMFs. Due to the different empiquencies in the different parts of the IMF, the subsampling should vary. The extrema points define the maximum empiquency in the IMF. Maximum empiquency is found by examining the space between these points. In the first IMF there are often areas where two neighbouring pixels both are extrema points, thus the maximum empiquency is 0.5. Letting this define our Nyquist bandwidth, we expect that it is not possible to subsample this IMF without distortion. Our suggestion is to treat the IMF block wise. This way the sampling rate for each block can be defined according to its empiquency content. The high empiquency blocks which cannot be subsampled according to our criterion are not modified. The remaining ones are subsampled.

Another proposal for variable sampling of an image is presented in [7], where an image coder based on non-uniform sampling of the image is presented. The sampling rate was determined by the complexity of the region. The use of Delaunay triangulation together with irregular sampling of an image is presented in Rila [72] in an image coding approach with promising results.

In this thesis we use overlapping blocks of size 7x7 pixels to minimize the artifacts from the blocking and to further reduce the samples used to represent the IMF. The corners of the block are always represented. The overlapping pixels in two neighbouring blocks will be the same, but used twice. This will ensure that the concatenated blocks have the same values at the edge pixels. The overlapping pixels will only belong to one of the blocks when patching together the reconstructed image.

In our EMD image coding proposals the image content steers the choice of method for each part of the image. This is the opposite of dynamic coding [27,24] where the selection is guided by the coding result. Dynamic coding deals with coding of images in which multiple coding schemes are used in different regions of an image.

11.4 EMD image coding

The image compression efforts in this work can be divided into two groups. The first group of methods use the full EMD, the image is decomposed down to a residue with only a few extrema points. The second group of methods use an EMD where only the first IMF is computed.

In the first group of methods we have:
Entropy coding of the EMD
Extrema Point Coder
Block-based DCT
Threshold coding of the full size DCT
VSDCTEMD

The first attempt to use the EMD for compression was the Extrema Point Coder [51]. This method is examined in this thesis both for time signals and images. The method of representing the IMF with its extrema points gives good visual performance but needs many extrema points for the representation of the first IMF and the coding of positions of these generate a high bitrate.

The block-based DCT is an attempt to use the property of the IMF that only one empiquency is present at each position. Using only one or two of the DCT components for each block in the representation gives bitrate and distortion measures that are acceptable but the visual performance suffers from severe blocking artifacts.

The Threshold DCT method works on the whole IMF and takes advantage of the energy compaction ability of the EMD. The energy of the IMFs other than the first IMF is concentrated in the low frequency regions and thus the rest of the frequency components can be left out of the representation of the EMD. Even for the first IMF this method gives acceptable distortion, both in terms of PSNR measure and visual performance, with bitrates below 2bpp. For the rest of the IMFs we have good performance at bitrates way below 1 bpp.

The variable sampling presents a promising method to treat the EMD. The structure of the variable sampling is inherited in the VSDCTEMD coding approach. The sampling process applied on each IMF and the residue leaves us with a reduced number of samples for each block. These can be squeezed into blocks of smaller sizes which can be DCT coded. The DCT components are then quantized and thresholded, leaving us with even fewer components to represent the block. The DCT coding of the block samples provides a significant compression of the variable sampled IMFs.

For image coding purposes there is no need to decompose the image into a full EMD. The second group of image compression methods use the empiquency of the first IMF to control the choice of coding method used in the different blocks of the image. The methods that we have developed are:

Entropy coded VS
VSDCTEMD+Threshold DCT
VSDCT

These new methods to code an image use the block-based variable sampling controlled by the first IMF. With the VS method we start with using the empiquency of the first IMF to control the variable sampling of the image. The image quality shown here is the best we can expect compared to the remaining coders in this section as they only differ in that they further compress the samples.

Threshold coding gives a very compact representation of the residue. Together with

the VSDCTEMD on the first IMF the compression result is very good at bitrates around 1 bpp.

The structure of the variable sampling is inherited in the VSDCT coding approach. The first IMF controls the sampling rate of the image block. The set of samples for each block left by the sampling process are squeezed into blocks of smaller size and DCT coded. The DCT components are then quantized and thresholded. For the VSDCT the compression result is also very good at bitrates around 1 bpp. This coder performs slightly better than the use of VSDCTEMD together with the Threshold DCT

The coders presented in this thesis are only early examples of image coding algorithms using the EMD. There is still much room for improvement and new ideas. A similar image compression method, but still totally different, is the multi-layered image representation of Meyer et al. [60], where the detail image is produced with a wavelet packet method, compressed and subtracted from the image to create a residue. This is iterated to produce subsequent residues. In this method the detail is contained in the residue images instead of in the IMFs. Apart from that difference it seems likely that new EMD compression algorithms will be influenced by the work on multi-layered coding.

11.5 Open problems

The work in this thesis was prompted by a central question: how to find, and use for compression, an adaptive decomposition of images. The work has found some answers but generated even more questions. Some of the problems raised from the work in this thesis but still unsolved are listed here.

- Develop the relation between the sequency and empiquency.
- Develop better algorithms for extrema point selection, that can handle saddle points.
- Use different sampling rates for horizontal and vertical in the variable sampling.
- Develop the compression algorithms further.
- Develop EMD for texture analysis.

Appendix A

In this appendix the 17 samples of Brodatz texture images used in these experiments are presented.

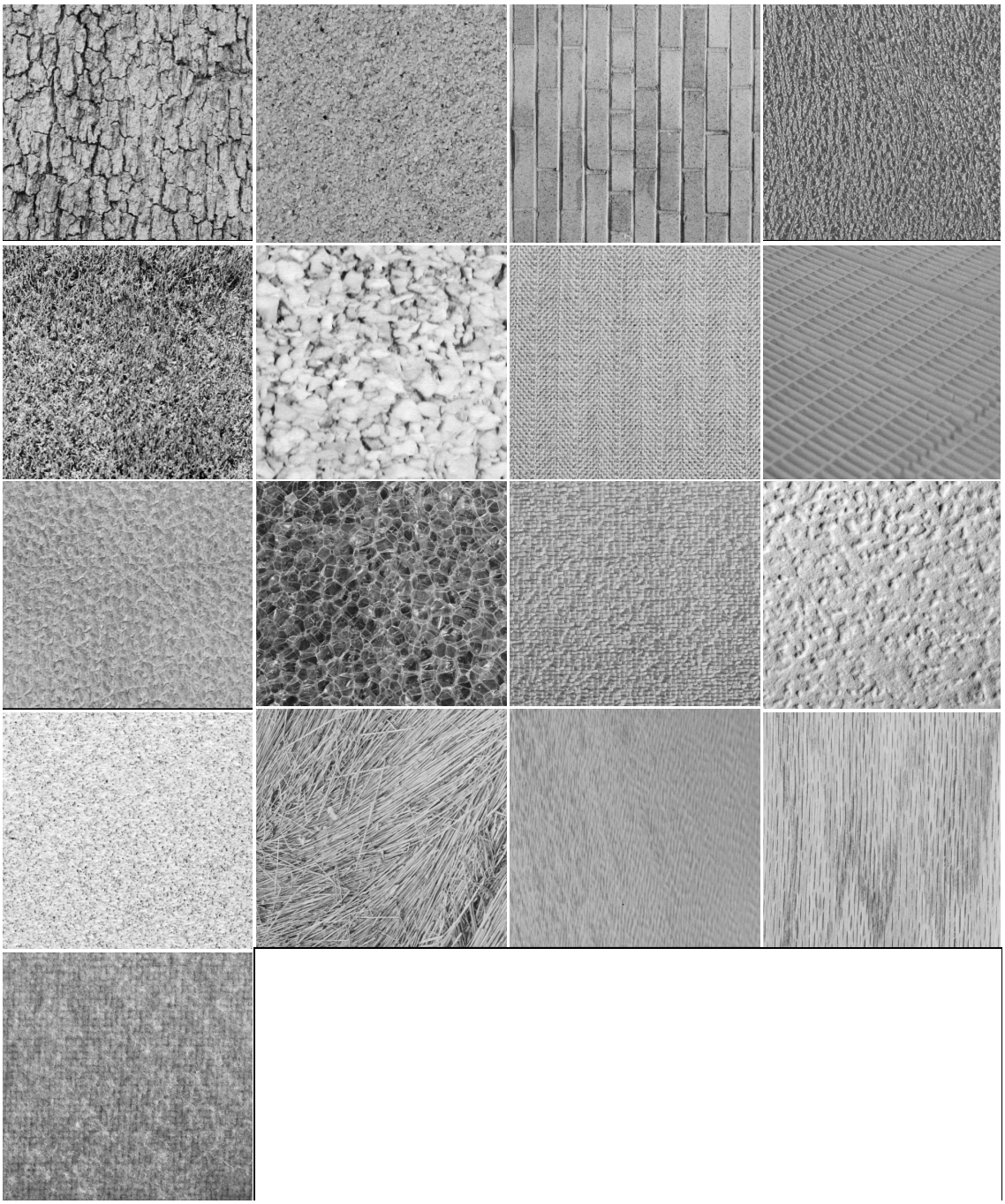


Figure A.1. The 17 samples of Brodatz texture images used in these experiments.

Appendix B

This appendix shows further parts of the triplet database. The textures in Figure B.1 are used as test images. Here, cost functions are varied for different typer of filters.

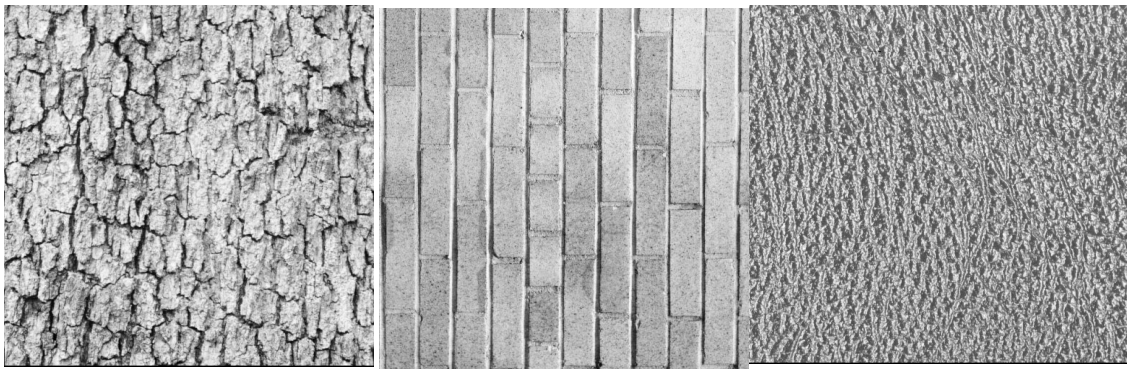


Figure B.1. The images bark, brickwall, cslfleath, from left to right.

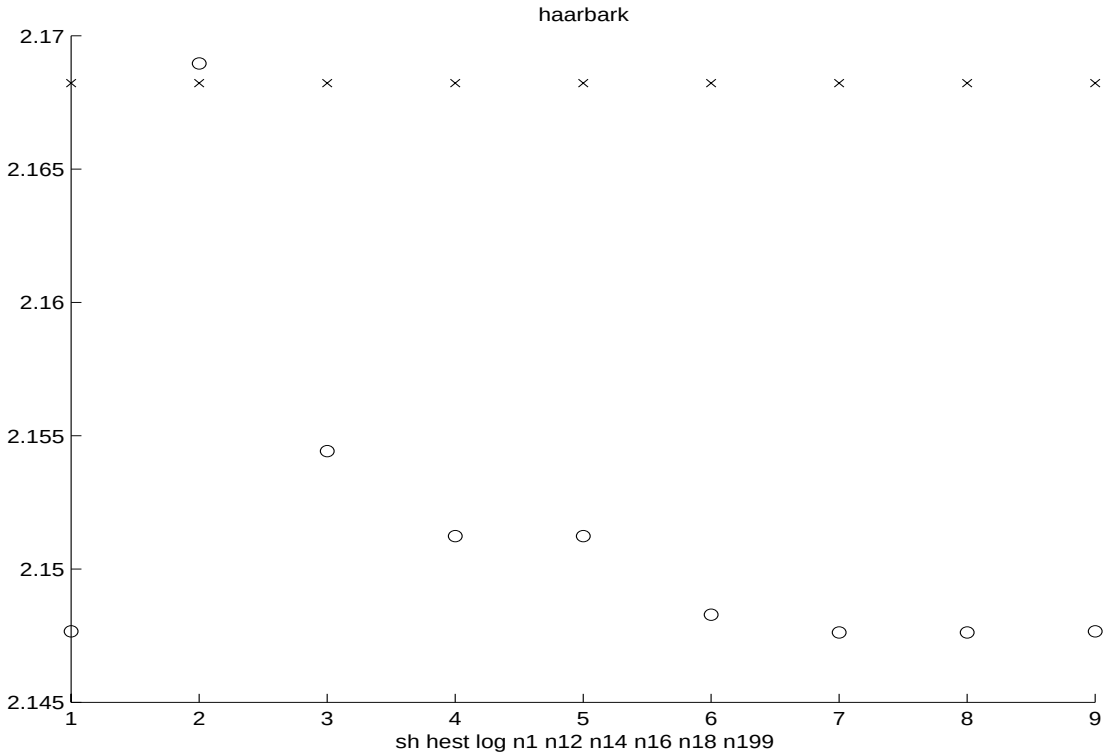


Figure B.2. The bark image decomposed with the Haar filter and the cost function is varied. The cost of coding the coefficients of the optimal wavelet packet basis is marked by **o** and the cost of coding the coefficients of the corresponding wavelet basis is marked by **x**.

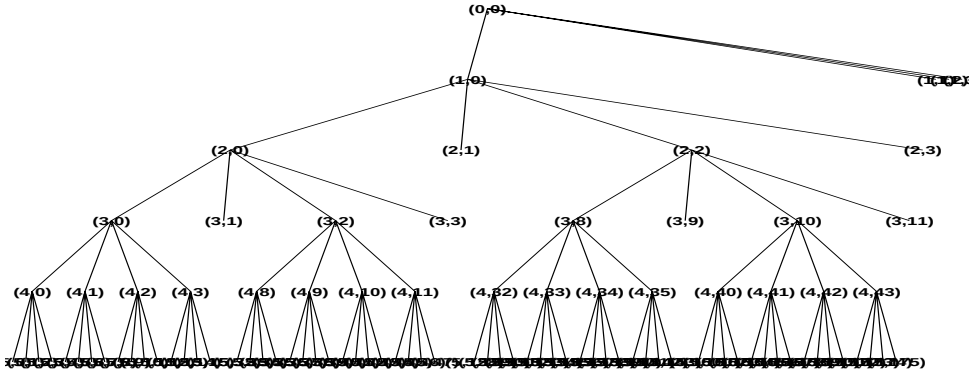


Figure B.3. The best basis for the triplet filter Haar, image bark, and cost function hest, defined in Eq. 4.42). The estimated entropy shows in position 2 in Figure B. 2.

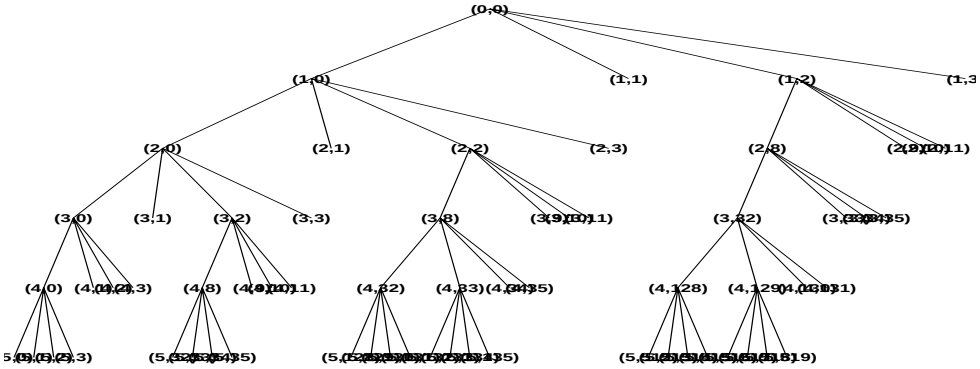


Figure B.4. The best basis for the triplet filter Haar, image bark, and cost function n1, defined in Eq. 4.44). The estimated entropy shows in position 4 in Figure B. 2.

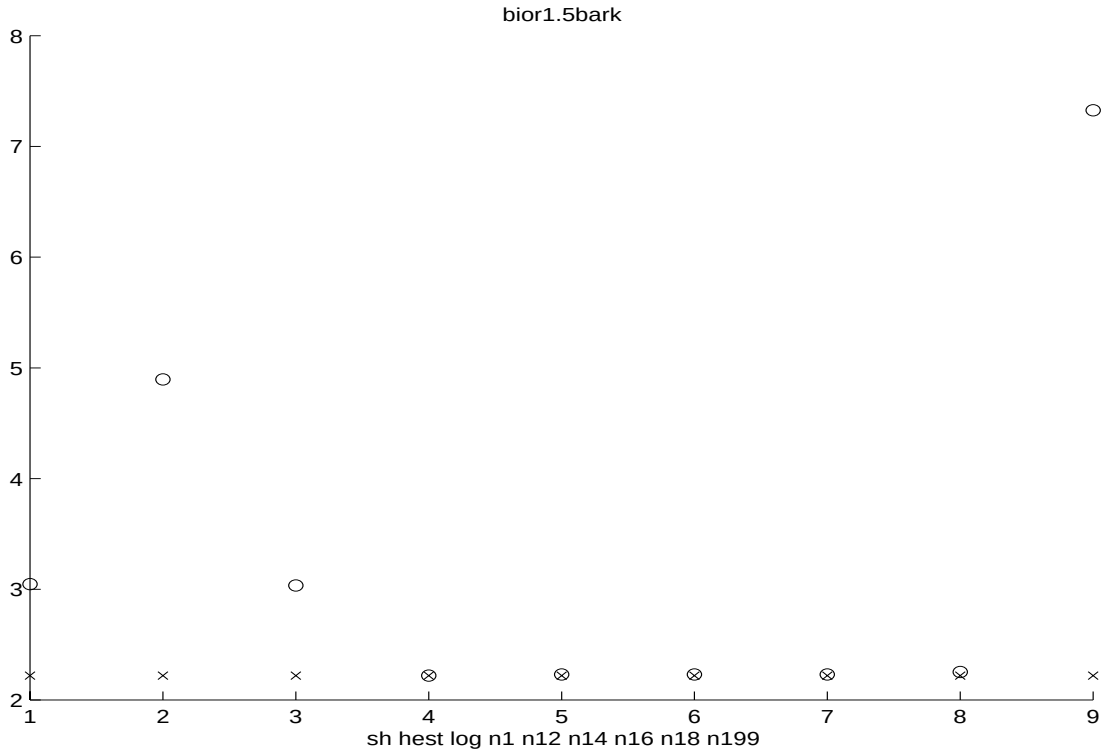


Figure B.5. The bark image decomposed with the filter bior1.5 and the cost function is varied. The cost of coding the coefficients of the optimal wavelet packet basis is marked by **o** and the cost of coding the coefficients of the corresponding wavelet basis is marked by **x**.

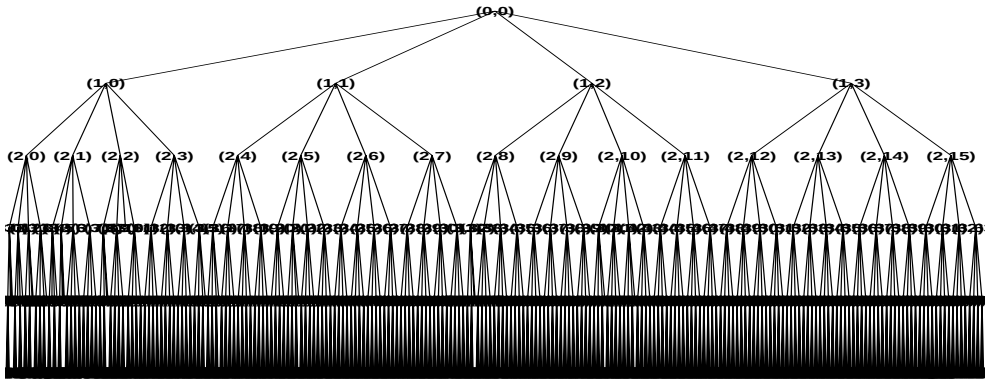


Figure B.6. The best basis for the triplet filter bior1.5, image bark, and cost function log, defined in Eq. 4.43).The estimated entropy shows in position 3 in Figure B. 5.

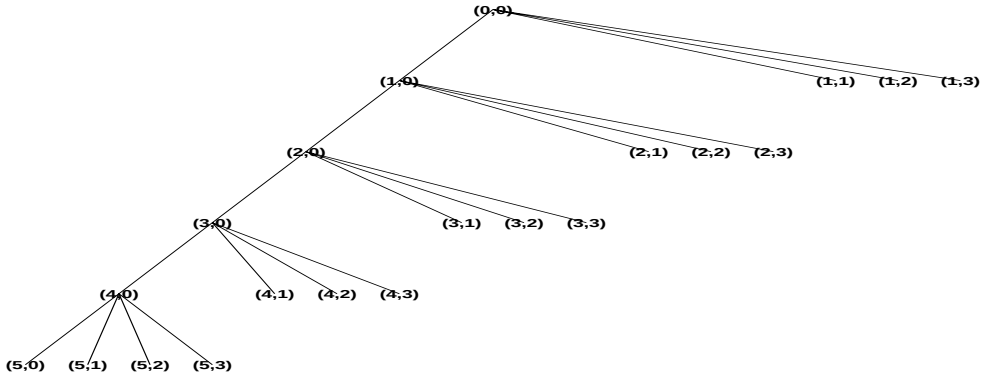


Figure B.7. The best basis for the triplet filter bior1.5, image bark, and cost function n1, defined in Eq. 4.44). The estimated entropy shows in position 4 in Figure B. 5.

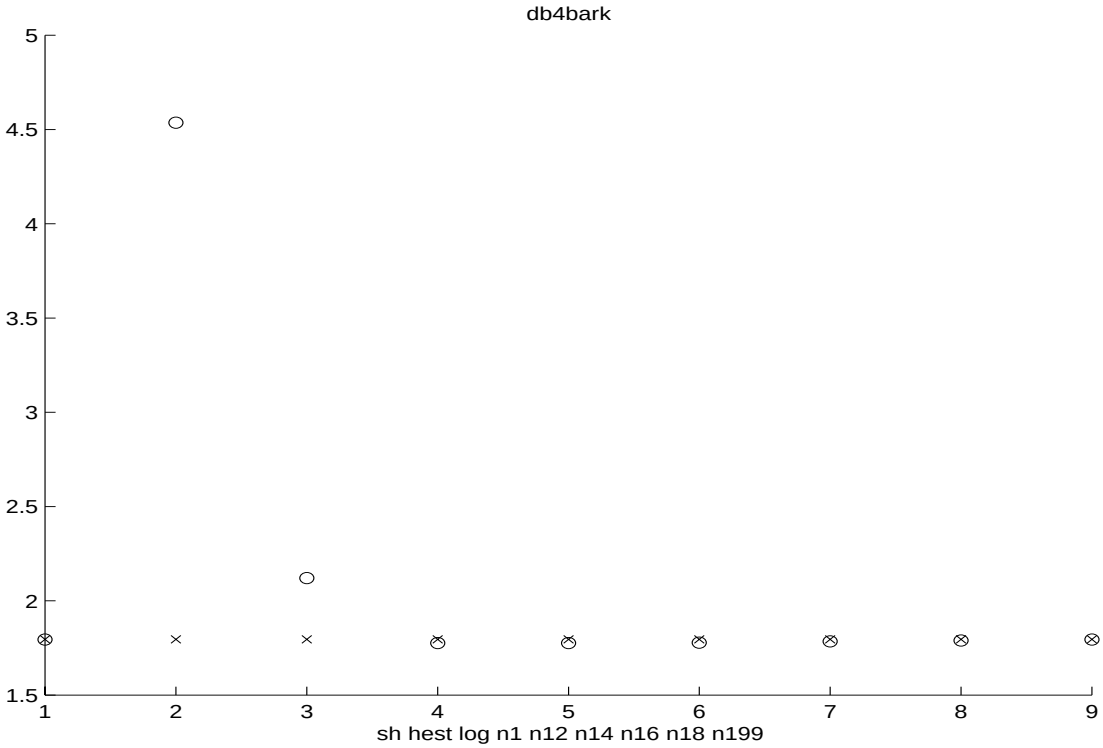


Figure B.8. The bark image decomposed with the db4 filter and the cost function is varied. The cost of coding the coefficients of the optimal wavelet packet basis is marked by **o** and the cost of coding the coefficients of the corresponding wavelet basis is marked by **x**.

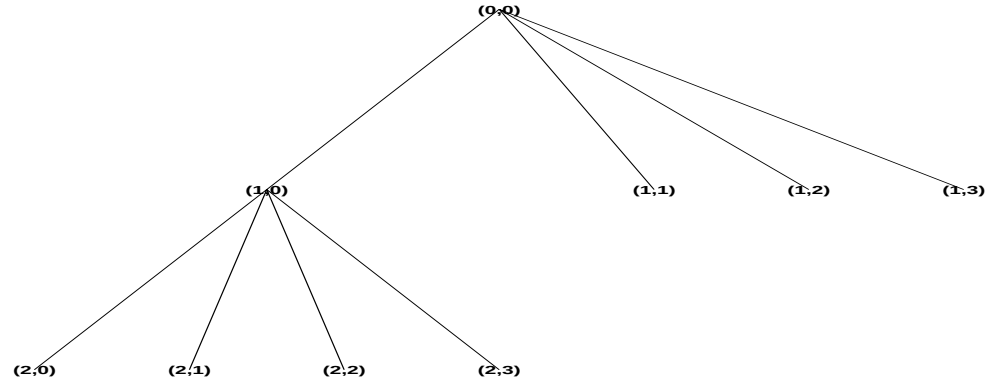


Figure B.9. The best basis for the triplet filter db4, image bark, and cost function hest, defined in Eq. 4.42). The estimated entropy shows in position 2 in Figure B. 8.

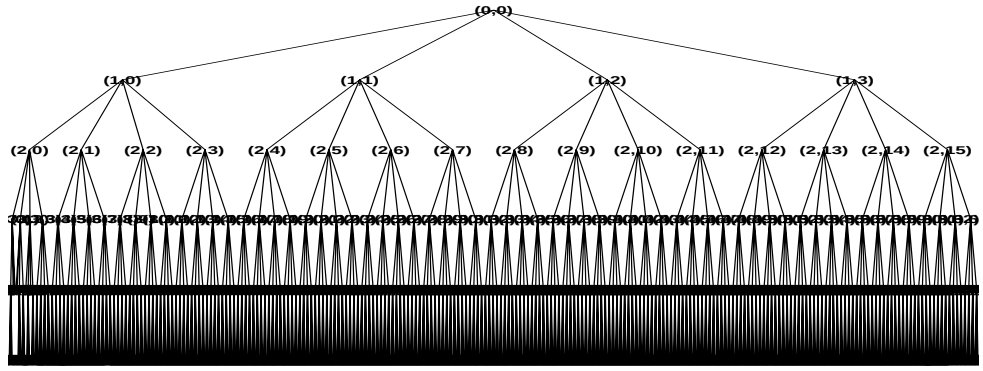


Figure B.10. The best basis for the triplet filter db4, image bark, and cost function log, defined in Eq. 4.43). The estimated entropy shows in position 3 in Figure B. 8.

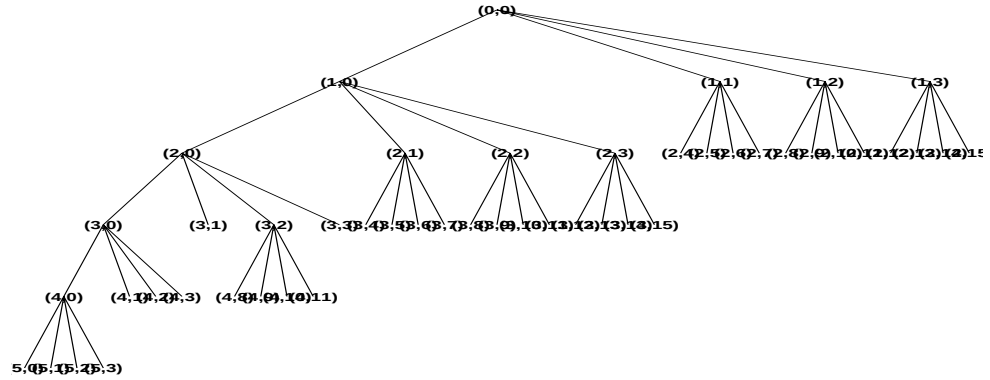


Figure B.11. The best basis for the triplet filter db4, image bark, and cost function n1, defined in Eq. 4.44). The estimated entropy shows in position 4 in Figure B. 8.

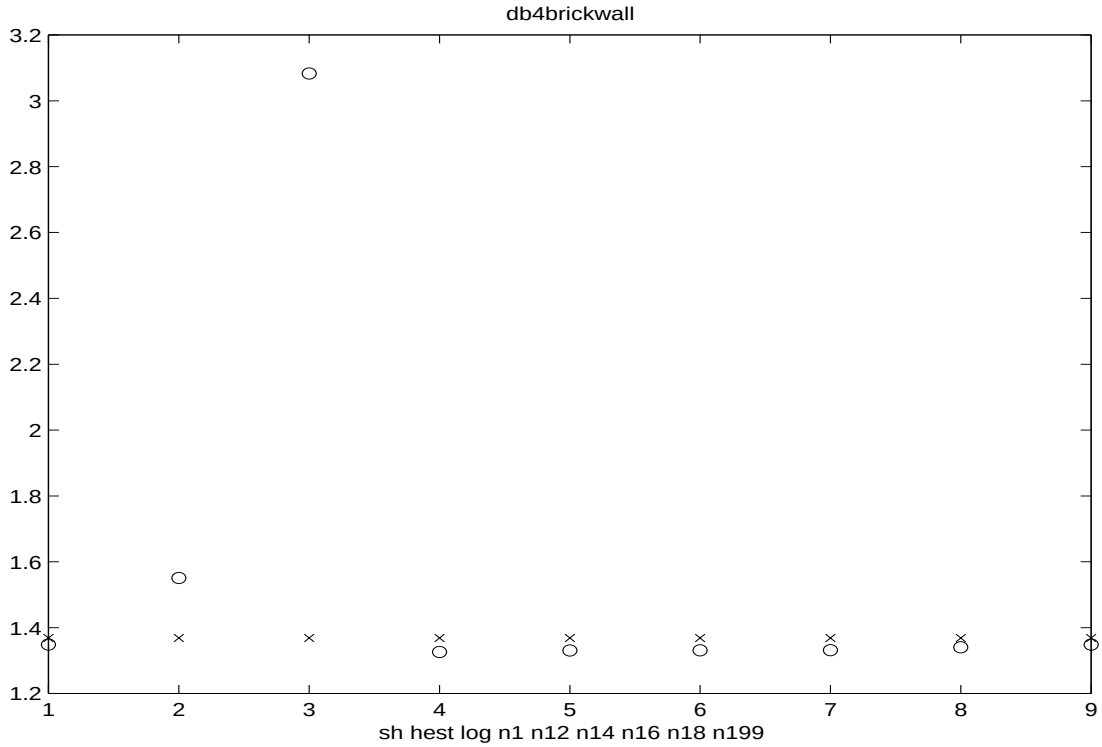


Figure B.12. The brickwall image decomposed with the db4 filter and the cost function is varied. The cost of coding the coefficients of the optimal wavelet packet basis is marked by **o** and the cost of coding the coefficients of the corresponding wavelet basis is marked by **x**.

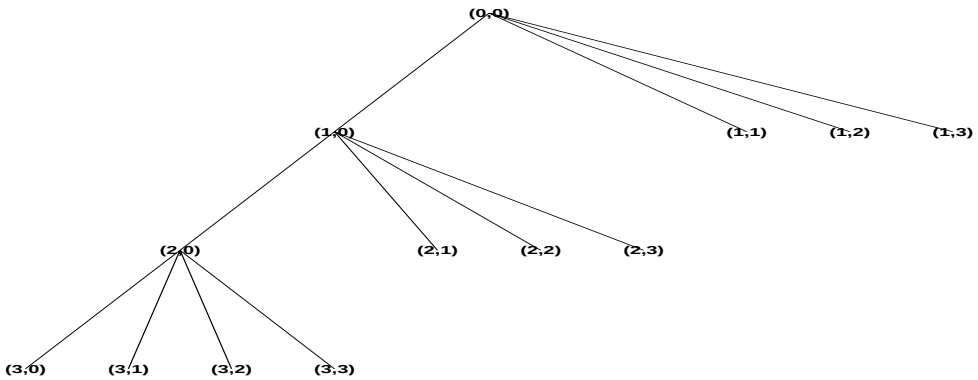


Figure B.13. The best basis for the triplet filter db4, image brickwall, and cost function hest, defined in Eq. 4.42). The estimated entropy shows in position 2 in Figure B. 12.

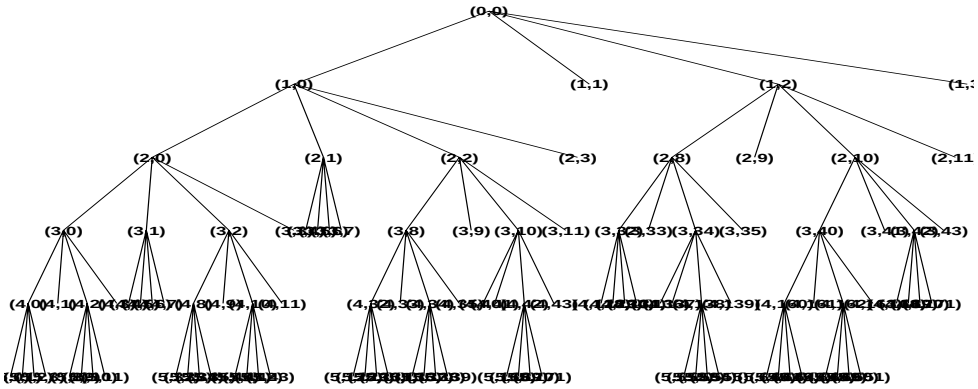


Figure B.14. The best basis for the triplet filter db4, image brickwall, and cost function n1, defined in Eq. 4.44). The estimated entropy shows in position 4 in Figure B. 12.

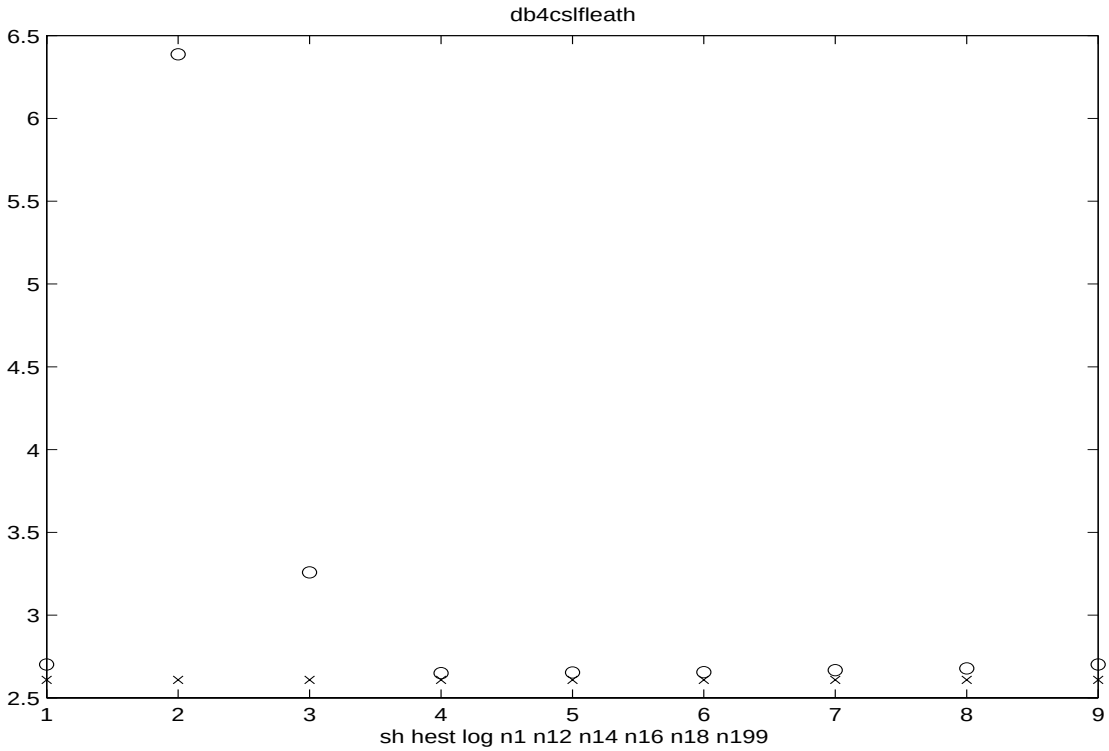


Figure B.15. The cslfleath image decomposed with the db4 filter and the cost function is varied. The cost of coding the coefficients of the optimal wavelet packet basis is marked by **o** and the cost of coding the coefficients of the corresponding wavelet basis is marked by **x**.

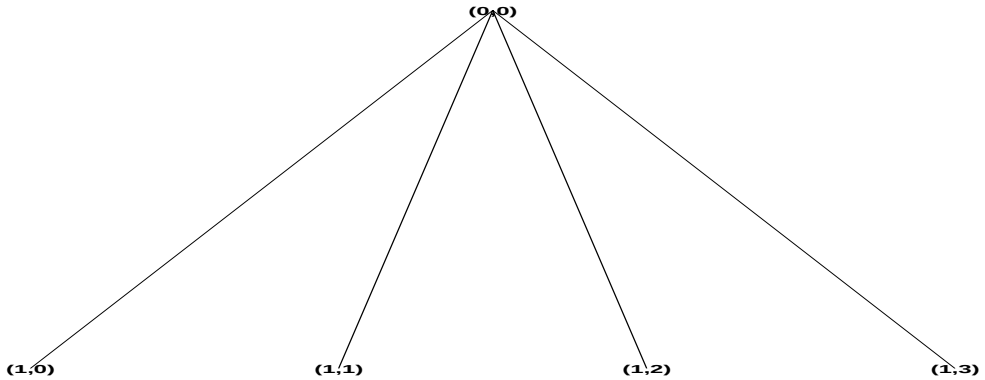


Figure B.16. The best basis for the triplet filter db4, image cslfleath, and cost function hest, defined in Eq. 4.42). The estimated entropy shows in position 2 in Figure B. 15.

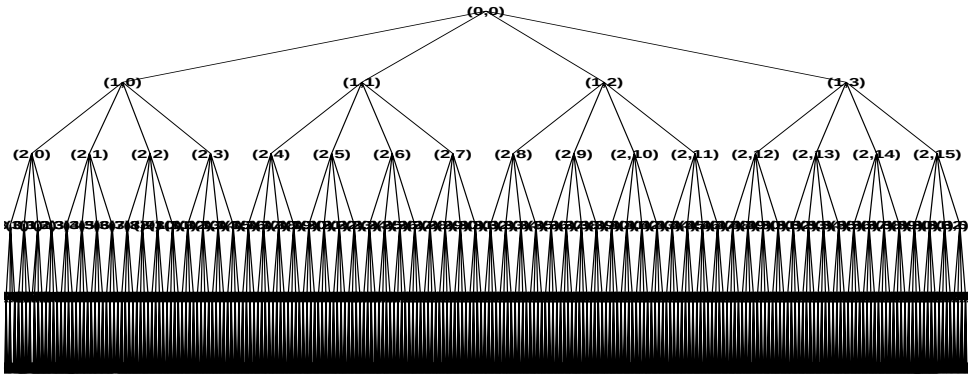


Figure B.17. The best basis for the triplet filter db4, image cslfleath, and cost function log, defined in Eq. 4.43). The estimated entropy shows in position 3 in Figure B. 15.

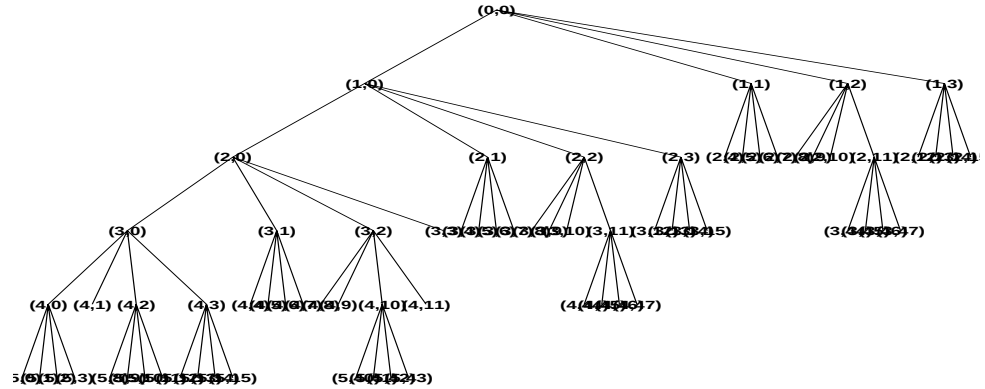


Figure B.18. The best basis for the triplet filter db4, image cslfleath, and cost function n1, defined in Eq. 4.44). The estimated entropy shows in position 4 in Figure B. 15.

Next, filters are varied for different cost functions to show filter effects.

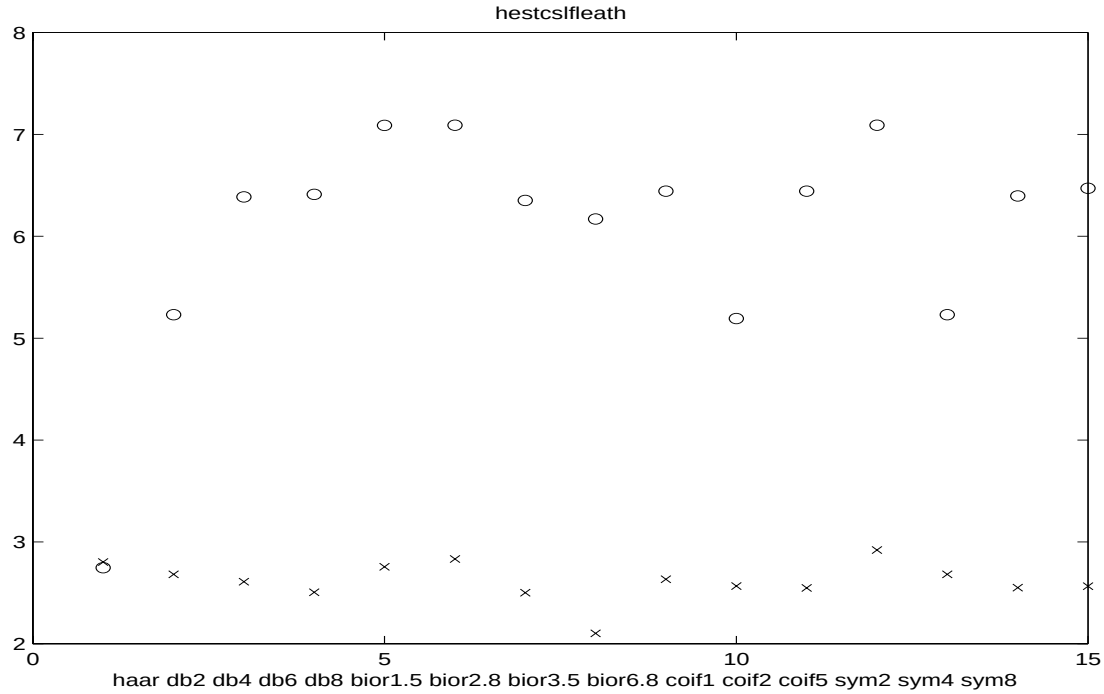


Figure B.19. The cslfleath image decomposed with varying filters and the cost function hest is used. The cost of coding the coefficients of the optimal wavelet packet basis is marked by **o** and the cost of coding the coefficients of the corresponding wavelet basis is marked by **x**.

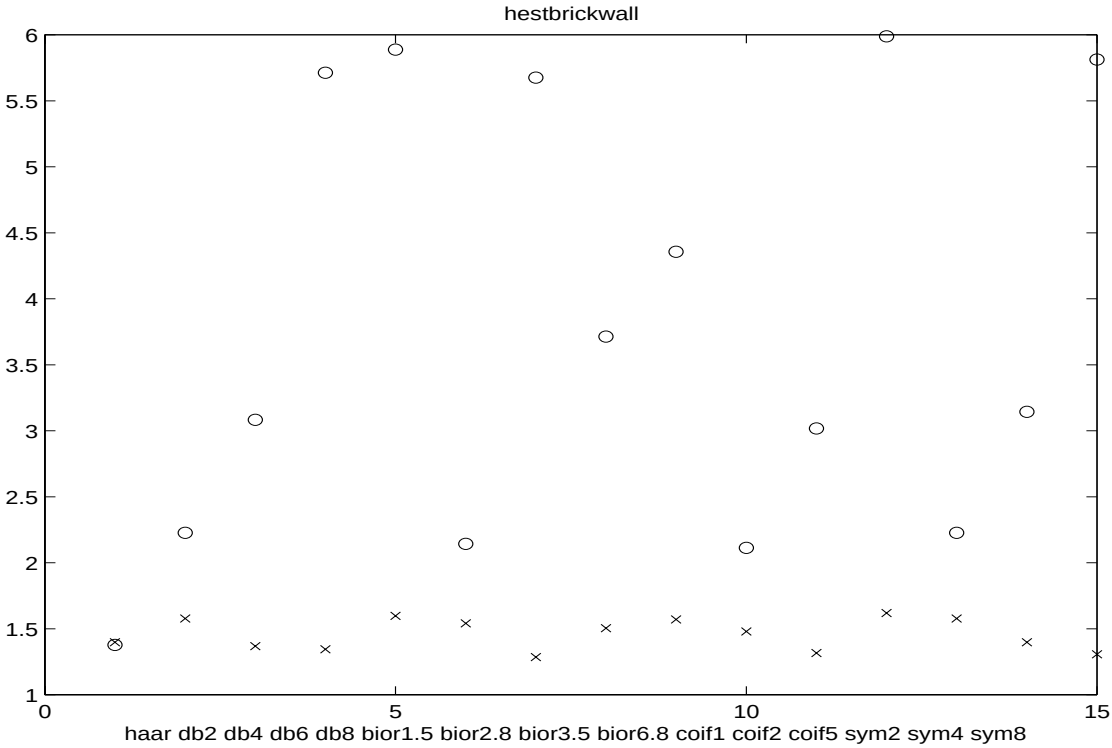


Figure B.20. The brickwall image decomposed with varying filters, using the cost function hest. The cost of coding the coefficients of the best basis is marked by **o** and the cost of coding the coefficients of the corresponding wavelet basis is marked by **x.**

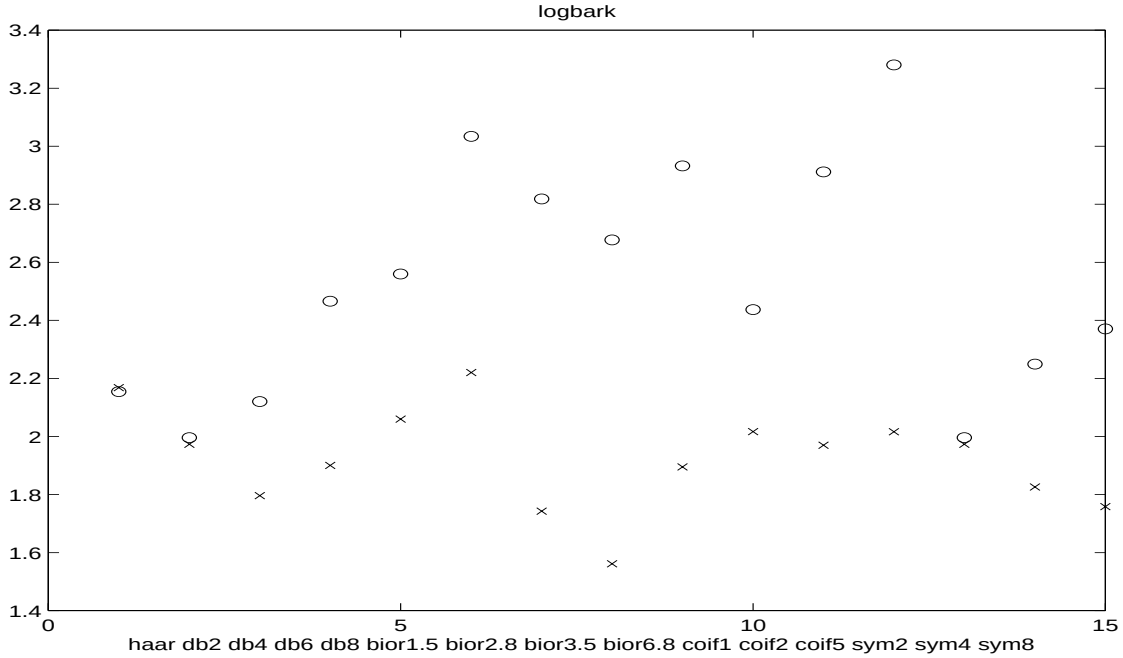


Figure B.21. The Bark image decomposed with varying filters and the cost function log is used. The cost of coding the coefficients of the optimal wavelet packet basis is marked by **o** and the cost of coding the coefficients of the corresponding wavelet basis is marked by **x.**

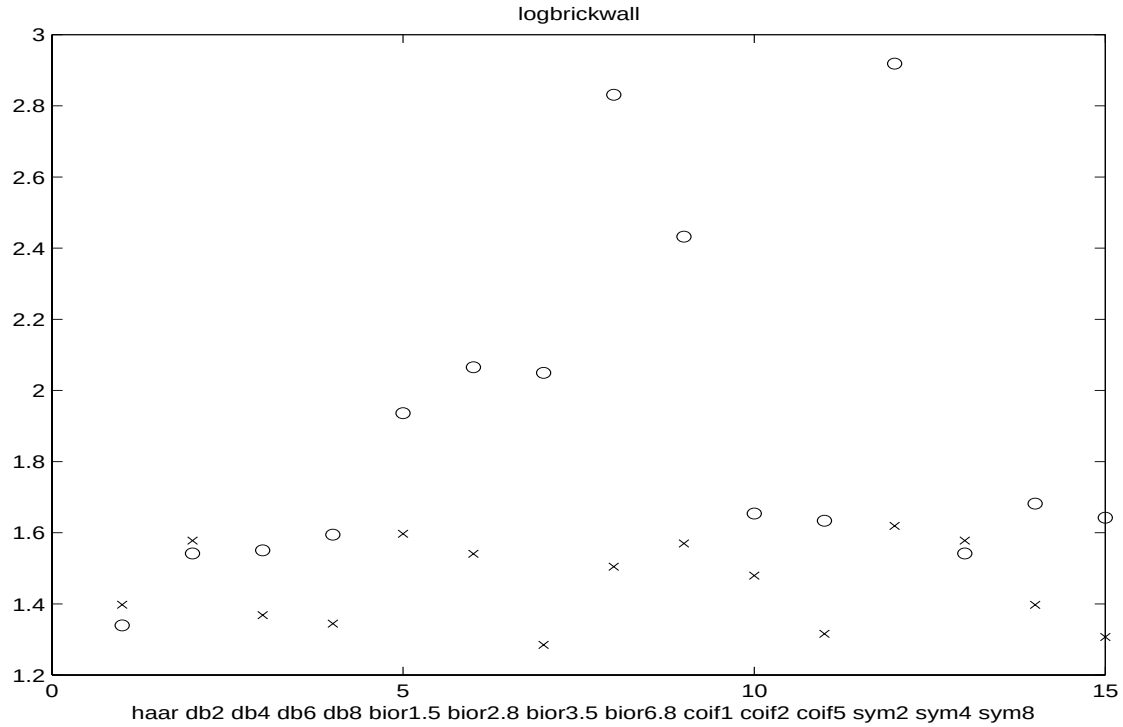


Figure B.22. The brickwall image decomposed with varying filters and the cost function log is used. The cost of coding the coefficients of the optimal wavelet packet basis is marked by **o** and the cost of coding the coefficients of the corresponding wavelet basis is marked by **x.**

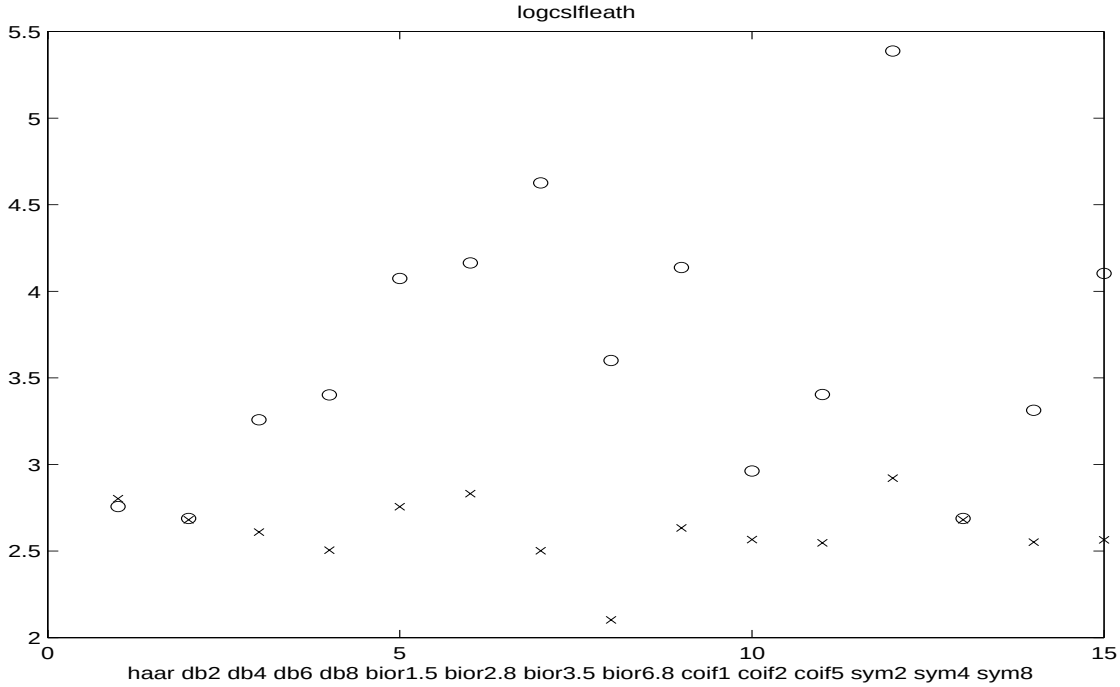


Figure B.23. The cslfleath image decomposed with varying filters and the cost function log is used. The cost of coding the coefficients of the optimal wavelet packet basis is marked by **o** and the cost of coding the coefficients of the corresponding wavelet basis is marked by **x.**

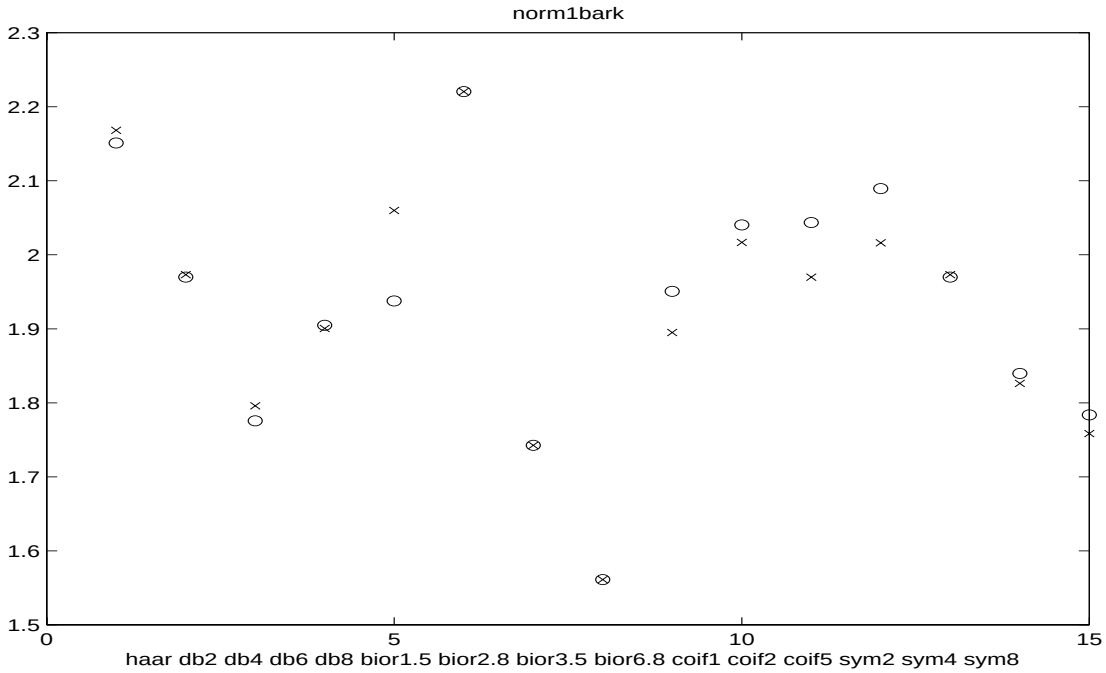


Figure B.24. The Bark image decomposed with varying filters and the cost function n1 is used. The cost of coding the coefficients of the optimal wavelet packet basis is marked by **o** and the cost of coding the coefficients of the corresponding wavelet basis is marked by **x**.

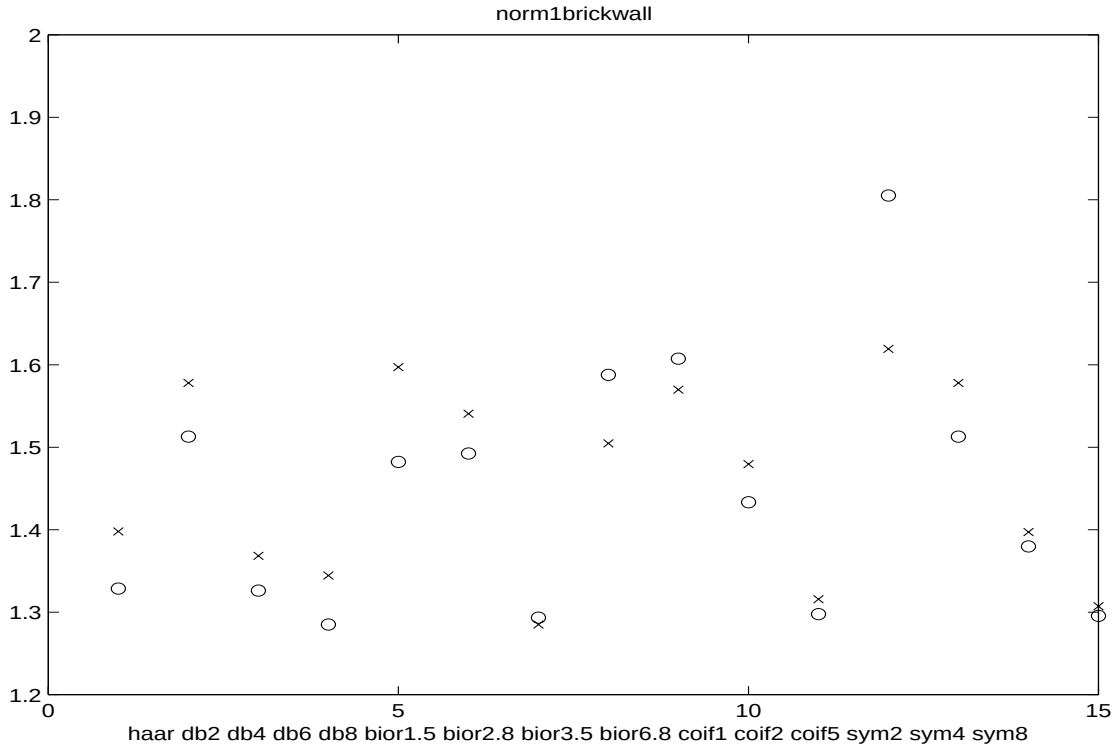


Figure B.25. The brickwall image decomposed with varying filters and the cost function n1 is used. The cost of coding the coefficients of the optimal wavelet packet basis is marked by **o** and the cost of coding the coefficients of the corresponding wavelet basis is marked by **x**

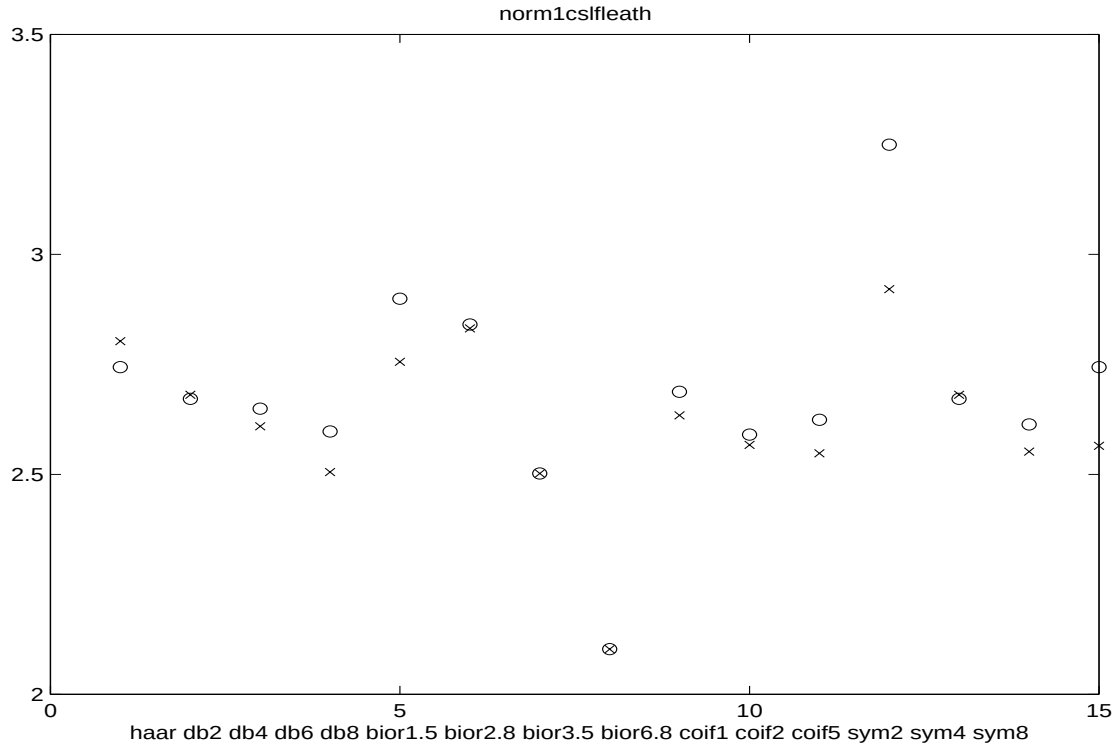


Figure B.26. The cslfleath image decomposed with varying filters and the cost function norm1 is used. The cost of coding the coefficients of the optimal wavelet packet basis is marked by **o** and the cost of coding the coefficients of the corresponding wavelet basis is marked by **x**

Appendix C

Listing of filter coefficients

'haar'

LO_D = 0.7071 0.7071

HI_D =-0.7071 0.7071

LO_R = 0.7071 0.7071

HI_R = 0.7071 -0.7071

'db2'

LO_D = -0.1294 0.2241 0.8365 0.4830

HI_D = -0.4830 0.8365 -0.2241 -0.1294

LO_R = 0.4830 0.8365 0.2241 -0.1294

HI_R = -0.1294 -0.2241 0.8365 -0.4830

'db4'

LO_D = -0.0106 0.0329 0.0308 -0.1870 -0.0280 0.6309 0.7148 0.2304

HI_D = -0.2304 0.7148 -0.6309 -0.0280 0.1870 0.0308 -0.0329 -0.0106

LO_R = 0.2304 0.7148 0.6309 -0.0280 -0.1870 0.0308 0.0329 -0.0106

HI_R = -0.0106 -0.0329 0.0308 0.1870 -0.0280 -0.6309 0.7148 -0.2304

'db6'

LO_D = -0.0011 0.0048 0.0006 -0.0316 0.0275 0.0975 -0.1298 -0.2263 0.3153 0.7511 0.4946 0.1115

HI_D = -0.1115 0.4946 -0.7511 0.3153 0.2263 -0.1298 -0.0975 0.0275 0.0316 0.0006 -0.0048 -0.0011

LO_R = 0.1115 0.4946 0.7511 0.3153 -0.2263 -0.1298 0.0975 0.0275 -0.0316 0.0006 0.0048 -0.0011

HI_R = -0.0011 -0.0048 0.0006 0.0316 0.0275 -0.0975 -0.1298 0.2263 0.3153 -0.7511 0.4946 -0.1115

'db8'

LO_D = -0.0001 0.0007 -0.0004 -0.0049 0.0087 0.0140 -0.0441 -0.0174 0.1287 0.0005 -0.2840 -0.0158 0.5854 0.6756 0.3129 0.0544

HI_D = -0.0544 0.3129 -0.6756 0.5854 0.0158 -0.2840 -0.0005 0.1287 0.0174 -0.0441 -0.0140 0.0087 0.0049 -0.0004 -0.0007 -0.0001

LO_R = 0.0544 0.3129 0.6756 0.5854 -0.0158 -0.2840 0.0005 0.1287 -0.0174 -0.0441 0.0140 0.0087 -0.0049 -0.0004 0.0007 -0.0001

HI_R = -0.0001 -0.0007 -0.0004 0.0049 0.0087 -0.0140 -0.0441 0.0174 0.1287 -0.0005 -0.2840 0.0158 0.5854 -0.6756 0.3129 -0.0544

'bior1.5'

LO_D = 0.0166 -0.0166 -0.1215 0.1215 0.7071 0.7071 0.1215 -0.1215 -0.0166 0.0166

HI_D = 0 0 0 0 -0.7071 0.7071 0 0 0 0

LO_R = 0 0 0 0 0.7071 0.7071 0 0 0 0

HI_R = 0.0166 0.0166 -0.1215 -0.1215 0.7071 -0.7071 0.1215 0.1215 -0.0166 -0.0166

'bior2.8'

LO_D = 0 0.0015 -0.0030 -0.0129 0.0289 0.0530 -0.1349 -0.1638 0.4626 0.9516 0.4626 -0.1638 -0.1349 0.0530 0.0289 -0.0129 -0.0030 0.0015

HI_D = 0 0 0 0 0 0 0 0.3536 -0.7071 0.3536 0 0 0 0 0 0 0 0

LO_R = 0 0 0 0 0 0 0 0.3536 0.7071 0.3536 0 0 0 0 0 0 0 0

HI_R = 0 -0.0015 -0.0030 0.0129 0.0289 -0.0530 -0.1349 0.1638 0.4626 -0.9516 0.4626 0.1638 -0.1349 -0.0530 0.0289 0.0129 -0.0030 -0.0015

'bior3.5'

LO_D = -0.0138 0.0414 0.0525 -0.2679 -0.0718 0.9667 0.9667 -0.0718 -0.2679 0.0525 0.0414 -0.0138

HI_D = 0 0 0 0 -0.1768 0.5303 -0.5303 0.1768 0 0 0 0

LO_R = 0 0 0 0 0.1768 0.5303 0.5303 0.1768 0 0 0 0

HI_R = -0.0138 -0.0414 0.0525 0.2679 -0.0718 -0.9667 0.9667 0.0718 -0.2679 -0.0525 0.0414 0.0138

'bior6.8'

LO_D = 0 0.0019 -0.0019 -0.0170 0.0119 0.0497 -0.0773 -0.0941 0.4208 0.8259 0.4208 -0.0941 -0.0773 0.0497 0.0119 -0.0170 -0.0019 0.0019

HI_D = 0 0 0 0.0144 -0.0145 -0.0787 0.0404 0.4178 -0.7589 0.4178 0.0404 -0.0787 -0.0145 0.0144 0 0 0 0

LO_R = 0 0 0 0.0144 0.0145 -0.0787 -0.0404 0.4178 0.7589 0.4178 -0.0404 -0.0787 0.0145 0.0144 0 0 0 0

HI_R = 0 -0.0019 -0.0019 0.0170 0.0119 -0.0497 -0.0773 0.0941 0.4208 -0.8259 0.4208 0.0941 -0.0773 -0.0497 0.0119 0.0170 -0.0019 -0.0019

'coif1'

LO_D = -0.0157 -0.0727 0.3849 0.8526 0.3379 -0.0727

HI_D = 0.0727 0.3379 -0.8526 0.3849 0.0727 -0.0157

LO_R = -0.0727 0.3379 0.8526 0.3849 -0.0727 -0.0157

HI_R = -0.0157 0.0727 0.3849 -0.8526 0.3379 0.0727

'coif2'

LO_D = -0.0007 -0.0018 0.0056 0.0237 -0.0594 -0.0765 0.4170 0.8127 0.3861 -0.0674 -0.0415 0.0164

HI_D = -0.0164 -0.0415 0.0674 0.3861 -0.8127 0.4170 0.0765 -0.0594 -0.0237 0.0056

0.0018 -0.0007

LO_R = 0.0164 -0.0415 -0.0674 0.3861 0.8127 0.4170 -0.0765 -0.0594 0.0237 0.0056 -0.0018 -0.0007

HI_R = -0.0007 0.0018 0.0056 -0.0237 -0.0594 0.0765 0.4170 -0.8127 0.3861 0.0674 -0.0415 -0.0164

'coif5'

LO_D = 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0001 0.0003 -0.0006 -0.0017 0.0024 0.0068 -0.0092 -0.0198 0.0327 0.0413 -0.1056 -0.0620 0.4380 0.7743 0.4216 -0.0520 -0.0919 0.0282 0.0234 -0.0101 -0.0042 0.0022 0.0004 -0.0002

HI_D = 0.0002 0.0004 -0.0022 -0.0042 0.0101 0.0234 -0.0282 -0.0919 0.0520 0.4216 -0.7743 0.4380 0.0620 -0.1056 -0.0413 0.0327 0.0198 -0.0092 -0.0068 0.0024 0.0017 -0.0006 -0.0003 0.0001 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

LO_R = -0.0002 0.0004 0.0022 -0.0042 -0.0101 0.0234 0.0282 -0.0919 -0.0520 0.4216 0.7743 0.4380 -0.0620 -0.1056 0.0413 0.0327 -0.0198 -0.0092 0.0068 0.0024 -0.0017 -0.0006 0.0003 0.0001 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

HI_R = 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0001 -0.0003 -0.0006 0.0017 0.0024 -0.0068 -0.0092 0.0198 0.0327 -0.0413 -0.1056 0.0620 0.4380 -0.7743 0.4216 0.0520 -0.0919 -0.0282 0.0234 0.0101 -0.0042 -0.0022 0.0004 0.0002

'sym2'

LO_D = -0.1294 0.2241 0.8365 0.4830

HI_D = -0.4830 0.8365 -0.2241 -0.1294

LO_R = 0.4830 0.8365 0.2241 -0.1294

HI_R = -0.1294 -0.2241 0.8365 -0.4830

'sym4'

LO_D = -0.0758 -0.0296 0.4976 0.8037 0.2979 -0.0992 -0.0126 0.0322

HI_D = -0.0322 -0.0126 0.0992 0.2979 -0.8037 0.4976 0.0296 -0.0758

LO_R = 0.0322 -0.0126 -0.0992 0.2979 0.8037 0.4976 -0.0296 -0.0758

HI_R = -0.0758 0.0296 0.4976 -0.8037 0.2979 0.0992 -0.0126 -0.0322

'sym8'

LO_D = -0.0034 -0.0005 0.0317 0.0076 -0.1433 -0.0613 0.4814 0.7772 0.3644 -0.0519 -0.0272 0.0491 0.0038 -0.0150 -0.0003 0.0019

HI_D = -0.0019 -0.0003 0.0150 0.0038 -0.0491 -0.0272 0.0519 0.3644 -0.7772 0.4814 0.0613 -0.1433 -0.0076 0.0317 0.0005 -0.0034

LO_R = 0.0019 -0.0003 -0.0150 0.0038 0.0491 -0.0272 -0.0519 0.3644 0.7772 0.4814 -0.0613 -0.1433 0.0076 0.0317 -0.0005 -0.0034

HI_R = -0.0034 0.0005 0.0317 -0.0076 -0.1433 0.0613 0.4814 -0.7772 0.3644 0.0519 -0.0272 -0.0491 0.0038 0.0150 -0.0003 -0.001

References

[1] E. H. Adelson, P. J. Burt, “Image data compression with the Laplacian pyramid”, Proceedings of the Pattern Recognition and Information Processing Conference, Dallas, TX. 1981, pp. 218-223.

[2] E. H. Adelson, E. Simoncelli, “Orthogonal pyramid transforms for image coding”, SPIE Proceedings Visual Communications and Image Processing II, Vol. 854, 1987, pp. 50-58.

[3] M. Antonini, M. Barlaud, P. Mathieu, I. Daubechies, “Image coding using wavelet transform”, IEEE Transactions on Image Processing, Vol. 1, No. 2 , April 1992, pp. 205 -220.

[4] N. Atreas, J. J. Benedetto, C. Karanikas, “Local Sampling for Regular Wavelet and Gabor Expansions”, Journal of Sampling theory in signal and image processing, Vol. 2, No. 1, Jan 2003, pp. 1-24.

[5] R. G. Baraniuk, D. L. Jones, ”Shear madness: new orthonormal bases and frames using chirp functions”, IEEE Transactions on Signal Processing, Vol. 41, No. 12, Dec.1993, pp. 3543 - 3549.

[6] R. G. Baraniuk, D. L. Jones, “Wigner-based formulation of the chirplet transform”, IEEE Transactions on Signal Processing, Vol. 44, No. 12, Dec 1996, pp. 3129 - 3135.

[7] W. Bender, C. Rosenberg, “Image Enhancement Using Non-Uniform Sampling”, SPIE Image Handling and Reproduction Systems Integration, vol. 1460, Feb. 1991, pp. 59-70.

[8] F. L. Bookstein., “Principal warps: Thin-plate splines and the Decomposition of deformations”, IEEE Trans. on Pattern analysis and machine intelligence, Vol. 11, No. 6, June 1989, pp. 567-585.

[9] R. N. Bracewell, The Fourier Transform And Its Applications, Second edition, McGraw-Hill, 1986.

[10] P. Brodatz, Textures, Dover Publications Inc., New York, 1966.

[11] R. W. Buccigrossi, E. P. Simoncelli, “Image compression via joint statistical characterization in the wavelet domain”, IEEE Transactions on Image Processing, Vol. 8, No. 12, Dec. 1999, pp. 1688 -1701.

[12] P. J. Burt, E. H. Adelson, “The Laplacian Pyramid as a Compact Image Code”, IEEE Transactions on Communications, Vol. COM-31, No. 4, April 1983, pp. 532-540.

[13] C. Christopoulos, A. Skodras, T. Ebrahimi, “The JPEG2000 still image coding system: an overview”, IEEE Transactions on Consumer Electronics, Vol. 46, No. 4, Nov. 2000, pp. 1103 –1127.

[14] L. Cohen, Time-Frequency Analysis, Prentice Hall, Englewood Cliffs, NJ, 1995.

[15] R. R. Coifman, personal communication 1995

[16] R. R. Coifman, M. V. Wickerhauser, “Best-adapted wave packets bases”, Preprint Yale University, New Haven,CT, 1990. Available at http://www.math.yale.edu/pub/wavelets/papers/bestbase.tex.

[17] R. R. Coifman, M. V. Wickerhauser, “Entropy-based algorithms for best basis selection”, IEEE Transactions on Information Theory, Vol. 38, No. 2, March 1992, pp. 713 –718.

[18] T. M. Cover, J. A. Thomas, Elements of Information Theory, Wiley Series in Telecomunications, John Wiley & Sons, Inc., New York, 1991.

[19] I. Daubechies, Ten lectures on wavelets, SIAM, Philadelphia, Pensylvania, 1992.

[20] I. Daubechies, “Orthonormal Bases of Compactly Supported Wavelets”, Comm. of Pure and Applied Mathematics, Vol. XLI, 1988, pp. 909-996.

[21] I. Daubechies, “Orthonormal Bases of Compactly Supported Wavelets II. Variations on a Theme”, SIAM Journal on Mathematical Analysis, Vol. 24, 1993, pp. 499-519.

[22] W. B. Davenport, Probability and Random Processes, McGraw-Hill, 1970.

[23] M. N. Do, M. Vetterli, “Wavelet-based texture retrieval using generalized Gaussian density and Kullback-Leibler distance”, IEEE Transactions on Image Processing, Vol. 11, No. 2, Feb. 2002, pp. 146 - 158.

[24] T. Ebrahimi, M. Kunt, “Object-Based Video Coding”, in Handbook of Image and Video Processing, Academic Press, 2000, pp. 585-596.

[25] P. Elias, “Universal Codeword Sets and Representations of the Integers”, IEEE Transactions on Information Theory, Vol. IT-21, No. 2, March 1975, pp. 194-203.

[26] P. Flandrin, G. Rilling, P. Goncalves, “Empirical Mode Decomposition as a Filter Bank”, IEEE Signal Processing Letters, Vol. 11, No. 2, Feb. 2004, pp. 112-114.

[27] P. Fleury, “Dynamic Scheme Selection in Image Coding”, PhD thesis 1999.

[28] J. Froment, S. Mallat, “Second Generation Compact Image Coding with Wavelets”, Wavelets - A Tutorial in Theory and Applications, Academic Press 1992, pp. 655-678.

[29] R. C. Gonzales, R. E. Woods, Digital Image Processing, Second edition, Prentice hall, Upper Saddle River, NJ, 2002.

[30] G. H. Granlund, H. Knutsson, Signal Processing for Computer Vision, Kluwer Academic Press, 1995.

[31] H. F. Harmuth, Transmission of Information by Orthogonal Functions, Springer-Verlag, Berlin/Heidelberg, 1969.

[32] C. Herley, J. Kovacevic, K. Ramchandran, M. Vetterli, “Time-varying orthonormal tilings of the time-frequency plane”, IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP-93, Vol. 3, 1993, pp. 205 -208.

[33] C. Herley, J. Kovacevic, K. Ramchandran, M. Vetterli, “Tilings of the Time-Frequency Plane: Construction of arbitrary Orthogonal Bases and Fast Tiling Algorithms”, IEEE Transactions on Signal Processing, Vol. 41, No. 12, 1993, pp. 3341 - 3359.

[34] C. Herley, Z Xiong, K. Ramchandran, M. T. Orchard, “Joint space-frequency segmentation using balanced wavelet packet trees for least-cost image representation”, IEEE Transactions on Image Processing, Vol. 6, No. 9, Sept. 1997, pp. 1213 –1230.

[35] C. Herley, P. W. Wong, “Minimum rate sampling and reconstruction of signals with arbitrary frequency support”, IEEE Transactions on Information Theory, Vol. 45, No. 5, July 1999, pp. 1555 - 1564.

[36] N. E. Huang et al. “The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis”. Proceedings: Mathematical, Physical and Engineering Sciences , Vol. 454, No. 1971, March 1998 , pp. 903-995.

[37] Y. Huanyin et.al, “A SAR Interferogram Filter Based on the Empirical Mode Decomposition Method”, International Geosciences and Remote Sensing Symposium, IGARSS 2001, Sydney, NSW, Australia, 9-13 July 2000, pp. 2061-2063.

[38] J. Karlholm, M. Ulvklo, S. Nyberg, A. Lauberts, A. Linderhed, “A survey of methods for detection of extended ground targets in EO/IR imagery”, FOI 2003, Scientific report FOI-R--0892--SE, Linköping.

[39] J. Karlholm, M. Ulvklo, J. Nygårds, M. Karlsson, S. Nyberg, M. Bengtsson, L. Klasén, A. Linderhed, M. Elmqvis, “Target detection and tracking processing chain: a survey of methods with special reference to EO/IR sequences, FOA 2000, User report FOA-R--00-01767-408,616--SE, Linköping.

[40] S. Khalid, Introduction to data compression, Second edition, Academic Press, 2000.

[41] H. Khalil, A. Jacquin, C. Podilchuk, “Constrained wavelet packets for tree-structured video coding algorithms”, Proceedings. Data Compression Conference, DCC ’99, Snowbird, UT, 29-31 March1999, pp. 354 –363.

[42] M. Khalili, M. Eisel, ARC Final Public Report, FOI Report FOI-S-1282-SE, Linköping, 2004.

[43] W.-H. Kim, T. Nguyen, Y.-H. Hu, “Adaptive wavelet packet based image coding with optimal entropy-constrained lattice vector quantizer (ECLVQ)”, IEEE Transactions on Signal Processing, Vol. 46, No. 7, July 1998, pp. 2019 –2026.

[44] P. Lancaster, K. Salkauskas, Curve and Surface Fitting, An Introduction, Academic press, 1986.

[45] R. E. Learned, W. C. Karl, A. S. Willsky, “Wavelet packet based transient signal classification”, Proceedings of the IEEE-SP International Symposium on Time-Frequency and Time-Scale Analysis, Victoria, BC, Canada, 4-6 October 1992, pp. 109 – 112.

[46] A. S. Lewis, G. Knowles, “Image compression using the 2-D wavelet transform”, IEEE Transactions on Image Processing, Vol. 1, No. 2, April 1992, pp. 244 –250.

[47] A. Linderhed, “Cost function and filter influence on wavelet packet decompositions of texture images”, Proceedings of Symposium on Image Analysis, SSAB’00, Halmstad, Sweden, March 7-8, 2000, pp. 5-8.

[48] A. Linderhed, “Wavelet packet decompositions of texture images - Cost function and filter influence”, Proceedings Internal Workshop for Information Theory and Image Coding 2001, report LiTH-ISY-R-2345, Linköping University, 2001, pp. 49-73.

[49] A. Linderhed, M. Lundberg, S. Nyberg, S. Sjökvist, M. Uppsäll, “Analysis of Optical Measurements on Real Minefields”, SPIE’s 16 th International symposium on Aerospace/Defense Sensing, Simulation and Controls, Orlando, FL, 1-5 April, 2002, pp. 160-171.

[50] A. Linderhed, “Wavelet packet decompositions of texture images: analysis of cost functions, filter influences, and image models”, SPIE Proceedings Vol. 4738, Wavelet and Independent Component Analysis Applications IX, Orlando, FL, SPIE Proceedings Vol. 4738, April 2002, pp. 9-20.

[51] A. Linderhed, “2D empirical mode decompositions in the spirit of image compression”, Wavelet and Independent Component Analysis Applications IX Orlando, FL,, SPIE Proceedings Vol. 4738, April 2002, pp. 1-8.

[52] A. Linderhed, “Image compression based on empirical mode decomposition”. Uppsala, March 11-12, 2004, Proceedings of Symposium on Image Analysis, SSAB’04, Uppsala, Sweden, 11-12 March 2004 pp. 110-113.

[53] A. Linderhed, “Variable Sampling of the Empirical Mode Decomposition of Two-Dimensional Signals”, Special issue on "Sampling and Frames in Wavelet Theory and Time-Frequency Analysis" of the International Journal of Wavelets, Multiresolution and Information Processing. Forthcoming.

[54] S. G. Mallat, “A theory for multiresolution signal decomposition: the wavelet representation”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 11, No. 7, July 1989, pp. 674 –693.

[55] S. G. Mallat, “Multifrequency channel decompositions of images and wavelet models”, IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 37, No. 12, Dec. 1989, pp. 2091 - 2110.

[56] S. Mallat, A wavelet tour of signal processing, Second edition, Academic Press, San Diego, CA, 1998.

[57] M. B. Martin, A. E. Bell, “New Image Compression Techniques Using Multiwavelets And Multiwavelet Packets”, IEEE Transactions on Image Processing, Vol.10, No. 4, April 2001, pp. 500-510.

[58] F. Marvasti, ”Interpolation of lowpass signals at half the Nyquist rate”, IEEE Signal Processing Letters, Vol. 3, No. 2, Feb. 1996, pp. 42-43.

[59] F. Marvasti, “Nonuniform sampling theorems for bandpass signals at or below the Nyquist density”, IEEE Transactions on Signal Processing, Vol. 44, No. 3, March 1996, pp. 572 -576.

[60] F.G. Meyer, A.Z. Averbuch, J-O. Strömberg, R.R. Coifman, “Multi-layered image representation: Application to image compression”, International Conference on Image Processing, ICIP'98, Chicago, 4-7 October, 1998.

[61] I Nedgård, "A comparison of analysis methods for vehicle classification by laser vibrometry", FOI report in press.

[62] J. C. Nunes, O. Niang, Y. Bouaoune, E. Delechelle, Ph. Bunel, “Texture analysis based on the bidimensional empirical mode decomposition with gray-level Co-occurrence models”, Proceedings of the Seventh International Symposium on Signal Processing and Its Applications, 2003, Vol. 2, 1-4 July, 2003, pp. 633 - 635.

[63] J. C. Nunes, Y. Bouaoune, E. Delechelle, O. Niang, P. Bunel, “Image analysis by bidimensional empirical mode decomposition”, Image and Vision Computing, Vol. 21, No. 12, Nov. 2003, pp. 1019-1026.

[64] A. V. Oppenheim, R.W. Schafer, Discrete-time Signal Processing, Second Edition, Prentice-Hall, Upper Saddle River, New Jersey, 1999.

[65] W. B. Pennebaker, J. L. Mitchell. JPEG Still Image Data Compression Standard, Van Nostrand Reinhold, New York, 1993.

[66] T. E. Posch, “The wave packet transform (WPT) as applied to signal processing”, Proceedings of the IEEE-SP International Symposium on Time-Frequency and Time-Scale Analysis, Victoria, BC, Canada, 4-6 October 1992, pp. 143 –146.

[67] F. P. Preparata, M. I. Shamos, Computational geometry: An Introduction, Springer-Verlag New York, Inc., New York, 1985.

[68] N.M. Rajpoot, R.G. Wilson, F.G. Meyer, R.R. Coifman, “Adaptive Wavelet Packet Basis Selection for Zerotree Image Coding”, IEEE Transactions on Image Processing, Vol. 12, No. 12, Dec. 2003, pp.1460-1472.

[69] K. Ramchandran, M. Vetterli, “Best wavelet packet bases in a rate-distortion sense”, IEEE Transactions on Image Processing, Vol. 2, No. 2, April 1993, pp. 160 –175.

[70] K. Ramchandran, M. Vetterli, “Best wavelet packet bases using rate-distortion criteria”, Proceedings of the IEEE International Symposium on Circuits and Systems, ISCAS ’92, Vol. 2, May 1992, pp. 971 -974.

[71] K. Ramchandran, X. Zixiang, K. Asai, M. Vetterli, “Adaptive transforms for image coding using spatially varying wavelet packets”, IEEE Transactions on Image Processing, Vol. 5, No. 7, July 1996, pp. 1197 –1204.

[72] L. Rila, “Image Coding using Irregular Subsampling and Delaunay Triangulation”, International Symposium on Computer Graphics, Image Processing, and Vision, 1998, pp. 167-173.

[73] G. Rilling, P. Flandrin, P Goncalvès, “On Empirical Mode Decomposition and its Algorithms”, Workshop on Nonlinear Signal and Image Processing NSIP-03, Grado Italy, June 2003.

[74] A. Rosenfeld, A. C. Kak, Digital Picture Processing, Academic Press, London, 1976.

[75] A. Said, W. A. Pearlman, “A new, fast, and efficient image codec based on set partitioning in hierarchical trees”, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 6, No. 3, June 1996, pp. 243 –250.

[76] S. D. Servetto, K. Ramchandran, M. T. Orchard, “Image coding based on a morphological representation of wavelet data”, IEEE Transactions on Image Processing, Vol. 8, No. 9, Sept. 1999, pp. 1161 –1174.

[77] C. E. Shannon, “A Mathematical Theory of Communication”, The Bell System Technical Journal, Vol. 27, 1948, pp. 379-423, 623-565.

[78] J. M. Shapiro, “Embedded image coding using zerotrees of wavelet coefficients”, IEEE Transactions on Signal Processing, Vol. 41, No. 12, Dec. 1993, pp. 3445 –3462.

[79] S. Sjökvist, A. Linderhed, S. Nyberg, M. Uppsäll, D. Loyd, “Minefield temporal feature extraction supported by heat transfer modelling”, EUDEM2 SCOT, International Conference on Requirements and Technology for the Detection, Removal and Neutralization of Landmines and UXO, Brussel, Belgium, 15-18 September 2003, pp. 401-409.

[80] S. Sjökvist, M. Uppsäll, S. Nyberg, A. Linderhed, M. Lundberg, “Optical detection of land mines at FOI”, SPIE Proceedings on Subsurface and Surface Sensing Technologies and Applications III, Vol. 4491, San Diego, CA, 2001, pp. 20-30.

[81] S. Sjökvist, A. Linderhed, S. Nyberg, M. Uppsäll, “Diurnal IR Features for Minefield Detection”, SPIE Defence & Security Symposium, Detection and Remediation Technologies for Mines and Minelike Targets, Orlando, FL, 12-16 April, 2004.

[82] J. R. Smith, S. Chang, “Frequency and spatially adaptive wavelet packets”, International Conference on Acoustics, Speech, and Signal Processing, Vol. 4, 1995, pp. 2233 -2236.

[83] W. Sweldens, “The Lifting Scheme: A Construction of Second Generation Wavelets”, Technical Report, Dept. of Mathematics, University of South Carolina, 1995.

[84] C. Taswell, “Near-best basis selection algorithms with non-additive information cost functions”, Proceedings of the IEEE-SP International Symposium on Time-Frequency and Time-Scale Analysis, 25-28 Oct. 1994, pp. 13 - 16.

[85] C. Taswell, “Satisficing search algorithms for selecting near-best bases in adaptive tree-structured wavelet transforms”, IEEE Transactions on Signal Processing, Vol. 44, No. 10, Oct. 1996, pp. 2423 - 2438.

[86] M. Unser, “Spline, A Perfect Fit for Signal and Image Processing”, IEEE Signal Processing Magazine, Nov 1999, pp. 22-38.

[87] M. Uppsäll, S. Sjökvist, A. Linderhed, S. Nyberg, “Compilation of FOI scientiffic contribution in ARC”, FOI report 2003 (FOI-S--1077--SE), Linköping.

[88] K. Vasudevan, F. A. Cook, “Empirical mode skeletonization of deep crustal data: Theory and applications”, Journal of Geophysical Research, Vol. 105, No. B4, April 2000.

[89] M. Vetterli, “Multi-dimensional sub-band coding: some theory and algorithms” Signal Processing, No6, 1984, pp 97-112.

[90] M. Vetterli, J. Kovacevic, Wavelets and subband coding, Prentice Hall, Upper Saddle River, NJ, 1995

[91] J. D. Villasenor, B. Belzer, J. Liao, “Wavelet filter evaluation for image compression”, IEEE Transactions on Image Processing, Vol. 4, No. 8, Aug. 1995, pp. 1053 - 1060.

[92] M. V. Wickerhauser, “Picture Compression by Best-Basis Sub-band Coding”, Technical Report Yale University, New Haven, CT, 1990.

[93] M. V. Wickerhauser, Adapted wavelet analysis from theory to software. A K Peters, Ltd. 1994.

[94] J. Woods, S. O'Neil, “Subband coding of images”, IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 34, No. 5, Oct 1986, pp. 1278 - 1288.

[95] Z. Xiong, K. Ramchandran, M. T. Orchard, “Wavelet packet image coding using space-frequency quantization”, IEEE Transactions on Image Processing, Vol. 7 No. 6, June 1998, pp. 892 –898

[96] R. Öktem, L. Öktem, K. Egiazarian, “A wavelet packet transform based image coding algorithm”, IEEE Nordic Signal Processing Symposium, NORSIG 2000, Kolmården, Sweden, 13-15 June 2000, pp. 77-80.