Chapter 4

Wavelet-Based Image Processing

In this chapter we will first explain how digital images are represented inside a computer system. After that we will discuss some image processing operations that give superior results when implemented using wavelets, or that just fit within the wavelet framework. We implemented all these operations in the software library WAILI, which will be described in Chapter 5.

One of the most important image processing operations is image compression. Wavelets also can provide for this.

4.1 Digital Images

A grey scale image can be considered as a rectangle where every point in the image has a luminance in the continuous range from black to white. To store an image in a computer, we need a finite precision representation. This is done in two steps: discretization and quantization.

4.1.1 Discretization

First we need to discretize the image to a finite number of pixels. A pixel is a small rectangle with a constant luminance. As a result we obtain a matrix of pixels. The number of pixels per length unit is called the resolution. The higher the resolution the better the quality of the discretized image, but the more memory needed.

4.1.2 Quantization

Every pixel has a luminance in the continuous range from black to white. To obtain a finite precision representation we restrict the pixel values to a finite number of quantization levels. Continuous grey scale images are often quantized to 256 (8 bit) levels: black is 0, white by 255. Grey values between black and white are mapped to an integer number in the range 0 to 255. More quantization levels improve the quality of the quantized image, but again require more memory.

We end up with a discrete representation of the original grey scale image, consisting of a matrix of integer luminance values, which can be manipulated by a computer.

4.1.3 Color images

Color images can be represented by three matrices: one for each primary color component red (R), green (G) and blue (B). Using these three primary colors one can represent most of the visible colors.

However, the human eye is much more sensitive to luminance differences than to color differences. One often uses a different decomposition, similar to the color encoding scheme used in the color television system. Now a color image is represented by the following three matrices:

Luminance matrix Y: This is the luminance of the image, a weighted mean of the three primary color components:

$$Y = c_R \cdot R + c_G \cdot G + c_B \cdot B,$$

with $c_R + c_G + c_B = 1$ (typically $c_R = 0.3$, $c_G = 0.59$ en $c_B = 0.11$).

Chrominance matrix U: This is the red difference, i.e. the difference of the red color component and the luminance:

$$U = R - Y.$$

Chrominance matrix V: This is the blue difference, i.e. the difference of the blue color component and the luminance:

$$V = B - Y.$$

Thus the luminance matrix Y is a grey scale version of the original color image, while the two chrominance matrices U and V contain the color information. The choice of the red and blue difference (and not the green difference) as the chrominance matrices was influenced by the higher sensitivity of the human eye to green light. Other linear combinations of the 3 primary color components are also in common use.

The original primary color components R, G and B can be easily reconstructed from the luminance and chrominance matrices.

4.1.4 Generalization

To make a more general statement, a digital image contains one or more matrices (channels) of integer values, one for each color component. Note that channels are not limited to the traditional three color components. They can contain any spectral data. Each channel can have a different size, to accommodate for subsampled chrominance channels, which are commonly used in e.g. broadcast applications.

4.1.5 $L^*a^*b^*$

Other color representations are also possible. Non-linear color representations like $L^*a^*b^*$ take into account the non-linearity in the response of the human eye to light [38].

4.2 Wavelet Denoising

4.2.1 Thresholding

An important application domain of wavelet theory is denoising. In a classical wavelet-based noise reduction algorithm first a discrete wavelet transform is computed. Then, the input wavelet coefficients are manipulated and finally an inverse transform yields the result. The *manipulation* of a coefficient often depends on the *class* to which the coefficient belongs. Therefore the algorithm first classifies the coefficients before the actual manipulation. For the allocation of each coefficient to a class, the algorithm uses a *criterion*, which is often binary: it divides the coefficients into two classes. The first group contains important, regular coefficients, while the other group of coefficients is cataloged as "not essential and too much affected by noise".

The most straightforward procedure uses the *absolute value* of the wavelet coefficients as a regularity measure: a coefficient close to zero contains little information and is relatively strongly influenced by noise. This is true because we consider the high-pass bands only and because small local variations in a signal are usually caused by noise. The basic idea is the same as for compression algorithms: the method assumes that a regular image can be represented by a small number of large coefficients. Donoho e.a. [18] showed that this method has statistical optimality properties. The absolute value is however a local measure: every coefficient is considered separately. More sophisticated algorithms often use a more global criterion.

When using such a threshold procedure, all wavelet coefficients with absolute value below a certain threshold δ are classified as "noisy". The algorithm replaces them by zero. The coefficients above the threshold are shrunk with the same value δ. Figure 4.1 compares this "soft-thresholding" operation with the "hard-thresholding" alternative. The hard-thresholding procedure does not shrink

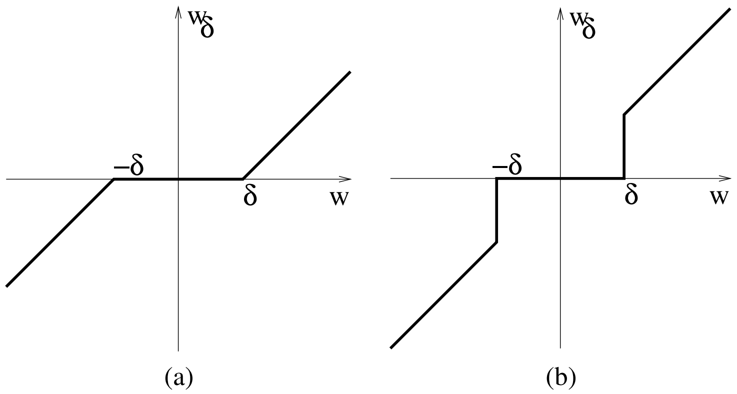

Figure 4.1: Noise reduction by wavelet shrinking. (a) Soft-thresholding: a wavelet coefficient w with an absolute value below the threshold δ is replaced by 0. Coefficients with higher absolute values are shrunk. (b) Hard-thresholding: Coefficients with an absolute value above the threshold are kept.

the coefficients with absolute value above the threshold. Although at first sight this may seem a more natural approach, soft-thresholding is a more continuous operation, and it is mathematically more tractable.

4.2.2 Threshold selection

The central issue in a threshold procedure is the selection of an appropriate threshold. If this threshold is too small, the result is still noisy. On the other hand, a large threshold also removes important image features and thus causes a bias. It is intuitively clear that the more noise is expected for a given coefficient, the higher the optimal threshold should be. Most procedures compute one threshold for a group of coefficients, based on the statistical properties of this group. Therefore, they assume additive, stationary noise within this group. If the input noise is additive and stationary, then it is easy to prove that the noise on the wavelet coefficients is additive and stationary within each subband and at each resolution level [27, 23]. This property does not hold true exactly for integer transforms, due to the non-linearity introduced by the rounding operations.

The optimal threshold for subband j $\delta_{j,\text{opt}}$ minimizes the mean square error of the result as compared with the unknown, noise-free coefficients:

$$R_j(\delta) = \frac{1}{N_j}\|w_{j,\delta} - v_j\|^2,$$

where N_j is the number of coefficients in subband j, $w_{j,\delta}$ is the vector of thresholded noisy coefficients and v_j is the vector of noise-free coefficients. The expected value of this error is called *Risk* function.

The "universal threshold" δ_j^U by Donoho and Johnstone [16] explicitly proposes a threshold value proportional to the amount of noise σ_j.

$$\delta_j^U = \sqrt{2\log N_j}\,\sigma_j.$$

In fact, for finite N_j this is an upper bound for the optimal threshold. This choice is asymptotically optimal: for infinitely long signals it is optimal, and thus the signal length has to be sufficiently large to obtain good results.

Another well known threshold selection procedure is based on "Stein's Unbiased Risk Estimator" (SURE) [17].

Based on the work of Jansen, we use the minimizer of the "Generalized Cross Validation" (GCV) function [60, 24, 23, 25]

$$GCV_j(\delta) = \frac{\frac{1}{N_j}\|w_j - w_{j,\delta}\|^2}{\left[\frac{N_{j0}}{N_j}\right]^2},$$

where N_{j0} is the number of wavelet coefficients that were replaced by zero. Unlike most other selection procedures it does not need an estimate of the actual noise level σ_j in subband j. Moreover, computational complexity is linear. The method is also asymptotically optimal, but for finite N_j it gives better estimates for the optimal threshold than δ_j^U.

Figure 4.2 illustrates the principle of generalized cross validation for a typical case. Artificial, stationary, colored noise was added to a clean test image. Both the noisy and noise-free image were transformed, using an integer version of the biorthogonal Cohen-Daubechies-Feauveau $(2,2)$ wavelet algorithm. At each level and for each subband, the GCV as a function of the threshold was compared with the exact mean square error. Although the conditions for a successful application of GCV are not strictly fulfilled by the non-linear character of the integer transform, both functions have approximately the same minimum. Note that in practical problems, a comparison with the exact mean square error is impossible, since the noise-free image is unknown.

4.2.3 Integer Minimization Problem

In a denoising implementation for a classical floating point wavelet transform the process of finding the threshold value that minimizes the GCV is a floating point minimization problem. Hence the minimization process is stopped when a given precision is reached.

However, in our case we use integer wavelets transform. Only a limited set of threshold values are meaningful due to the finite precision of the integer data type. This means that we can always calculate the exact (integer) threshold value that minimizes the GCV in a few iteration steps. In our implementation, we used an integer variant of the Fibonacci minimizer [20].

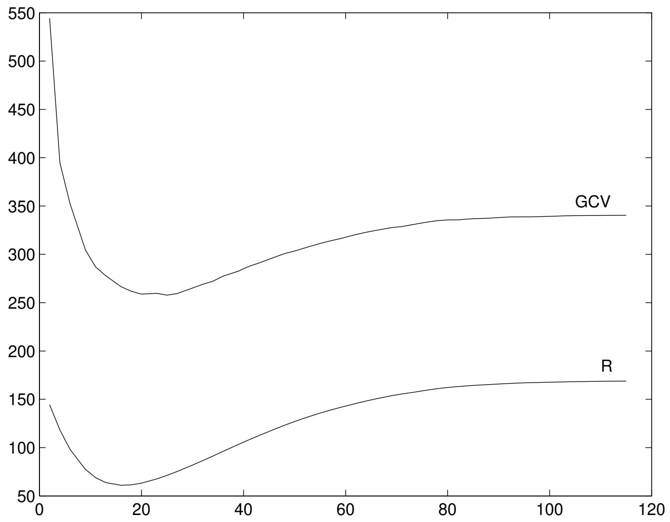

Figure 4.2: *GCV* and mean square error of the result in function of the threshold δ. Both functions have approximately the same minimum.

4.3 Image Scaling

After one step of a wavelet transform on an image, it is clearly visible that the low-pass subband is a downscaled version (by a factor of 2) of the original image. Thus a wavelet transform can be used to downscale an image with a factor of 2^n (Figure 4.3). Because of the low-pass filtering, no aliasing occurs.

Consequently, the inverse wavelet transform can be used to upscale an image with a factor of 2^n, if we consider all wavelet coefficients in the high-pass subbands to be zero[1]. This gives superior results than simple pixel replication (Figure 4.4), since no blocking effect occurs.

Hence we can down- or upscale an image by a factor of 2^n in the wavelet domain, by removing or adding subbands.

To scale an image with a factor that is not a power of two, one can scale it using a wavelet transform until it is larger than the desired image size, and use classical (linear) interpolation to downscale it until the image has the wanted size [62]. As a side note, upscaling an image using the CDF $(2,x)$ wavelet transform is equivalent to upscaling using linear interpolation (Figure 4.5). Other wavelet transforms correspond to other interpolation rules (e.g. CDF $(4,x)$ is equivalent to cubic in-

[1] It could be better to try to predict the coefficients in the high-pass subbands based on the values of the coefficients in the subbands at the lower resolution levels instead of simply setting them to zero, but this is much more difficult.

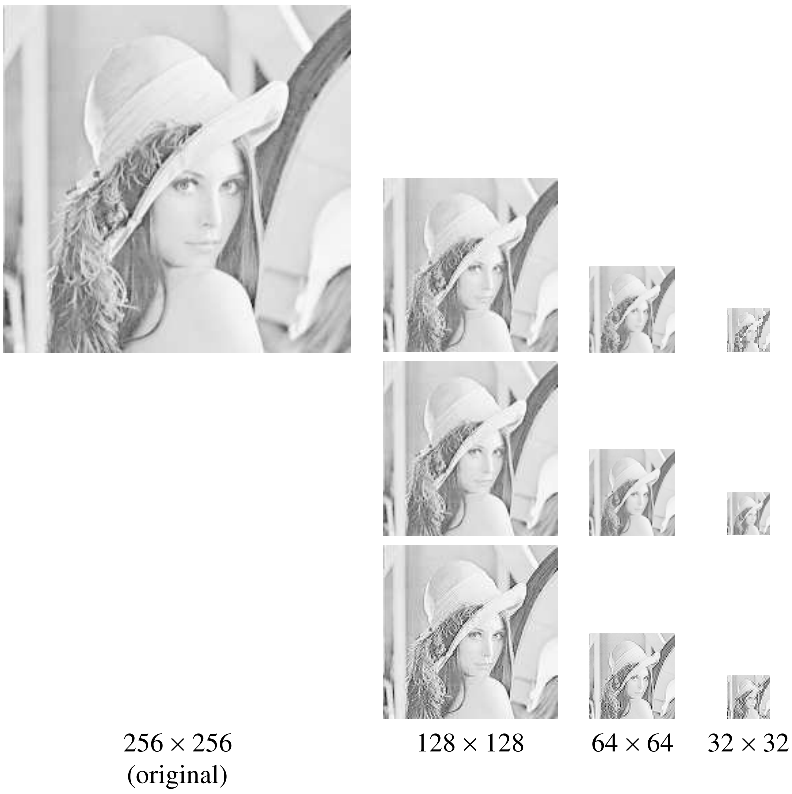

Figure 4.3: Downscaling of an image using simple subsampling (upper row) and using the CDF $(2,2)$ (middle row) and CDF $(4,4)$ wavelet transforms (lower row). Aliasing occurs with simple subsampling, while the wavelet-based method does not suffer from aliasing because the low-pass filter in a wavelet transform smoothes the image. These images were created using WAILI.

32×32 (original) 64×64 128×128 256×256

Figure 4.4: Upscaling of an image using pixel replication (upper row) and using the CDF $(2,2)$ (middle row) and CDF $(4,4)$ wavelet transforms (lower row). Pixel replication shows a clearly visible blocking effect, which is not present when using the wavelet-based method. These images were created using WAILI.

(a) (b)

Figure 4.5: Upscaling of an image with a factor 5 using (a) pixel replication and using (b) the CDF $(2,2)$ wavelet transform.

terpolation).

4.4 Crop and Merge

Accessing a rectangular part of an image is trivial in the spatial domain. Although wavelets are localized in the spatial domain, crop and merge operations on wavelet transformed images are much more difficult to implement because the wavelet transform is not translation invariant (see Section 2.4). Due to the importance of crop and merge operations we had to circumvent this problem, and our solution will be given in Section 5.3.1.

4.5 Image Compression

4.5.1 Why Compression?

The amount of digital images we use is growing continuously, and video and television transmission becomes more and more digital. Digital images become larger and larger, because there is an increased need for large scenes or because the required resolution increases. On the other hand, we use more and more digital image sequences in multi-media applications.

To illustrate the need for compression, we give some examples:

- To store a color image of a moderate size, e.g. 512×512 pixels, one needs 0.75 MB of disk space.
- A 35 mm slide digitized with a resolution of 12 μm requires 18 MB.
- One second of digital PAL video requires 27 MB.

- A digital orthophotomap of a Belgian province, with a resolution of one pixel per m^2 requires about 15 GB (Belgium takes 90 GB).

To store these images, and make them available over networks (e.g. the Internet), compression techniques are needed. The use of compression techniques also allows the processing of larger images using the current hardware, while the processing of images of a given size can be performed on cheaper hardware platforms.

4.5.2 Compression Techniques

Lossless versus Lossy Compression

Compression methods can be divided in two classes: lossless and lossy compression techniques:

- *Lossless compression* guarantees that the original signal can be reconstructed without any errors. This is important for applications like the compression of text. For images, lossless compression is often used as the second step, after the lossy part.
- With *lossy compression*, we can obtain higher compression rates by not requiring the exact data to be reconstructed. Indeed, because the human visual system is not sensitive to some kinds of errors, the compression potential is much higher when we allow for small reconstruction errors.

Although the integer wavelet transform can be used for lossless compression due to its 100% invertible nature (in contrast to floating point wavelet transform implementations), image compression will usually be lossy due to the high compression rates that are required, except in sensitive applications areas like medical imaging where any data loss is not acceptable.

A Lossy Compression Scheme

Lossy compression schemes are commonly based on transform coding. Such a coding scheme consists of three steps, as is shown in Figure 4.6.

Transform The original signal is transformed using a decorrelating transform. If one can represent the original signal f as

$$f = \sum_i a_i \psi_i,$$

using only a few basis functions ψ_i, one needs only the few coefficients a_i to reconstruct the signal f. Popular transforms are the Fast Fourier Transform (FFT) and its derivatives (e.g. the Discrete Cosine Transform (DCT)), and of course wavelet transforms.

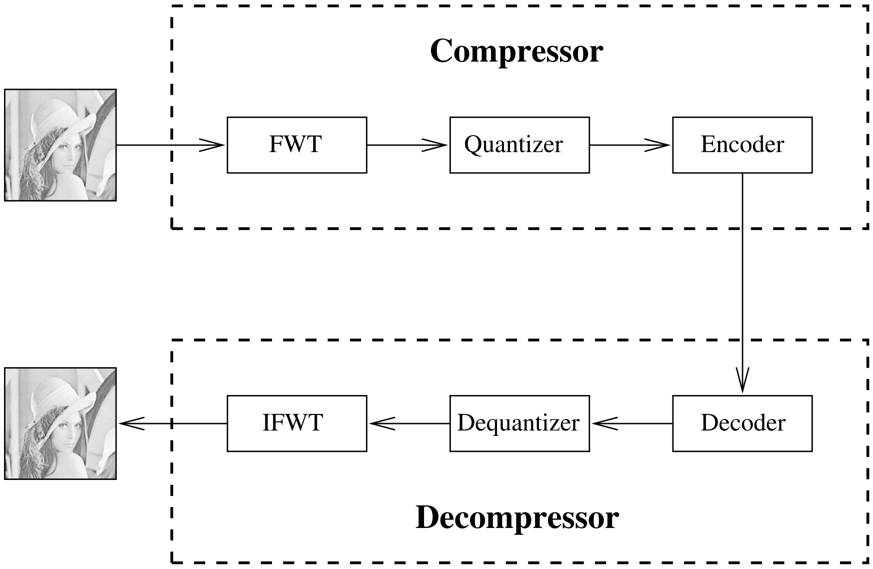

Figure 4.6: General scheme of a lossless image compression algorithm using wavelet transforms.

Quantization The quantization step defines which coefficients will be neglected, and with which accuracy the retained coefficients will be stored. More important coefficients will be represented with a higher accuracy. The least important coefficients (very close to zero) can be replaced by zero. This is the lossy step in a lossy compression scheme, if we assume the transform step is invertible without any loss.

Mathematically, Parseval's identity (Equation 2.4, in the case of an orthogonal basis), or the fact that we have a Riesz basis (Equation 2.5, in the biorthogonal case), guarantee that the largest coefficients are the most important coefficients. Neglecting a coefficient with a small magnitude gives a reconstruction error proportional to the magnitude of the coefficient.

Encoding Afterwards the quantized coefficients (symbols) will be stored in a bit stream in an efficient way. Usually one uses some sort of entropy encoding to convert the coefficients to a bit stream.

Color Images

Color images have additional compression potential because the human eye is less sensitive to changes in chrominance than to changes in luminance. Thus color images can be compressed with higher compression rates than the corresponding grey scale images.

4.5.3 Features of Image Compression using Wavelets

The key features of wavelet-based compression schemes are:

- Wavelets provide good compression ratios [15, 5]. Especially for high compression ratios, wavelets perform much better than competing technologies like JPEG [59], both in terms of signal-to-noise ratio and visual quality. Unlike JPEG, they show no blocking effect but allow for a graceful degradation of the whole image quality, while preserving the important details of the image. The next version of the JPEG standard (JPEG 2000) will incorporate wavelet based compression techniques.

- The wavelet transform is a fast operation (linear to the amount of data), especially when implemented using the lifting scheme. The wavelet transform is symmetric: both the forward and the inverse transform have the same complexity, allowing fast compression and decompression routines.

- Multi-resolution allows for progressive transmission and zooming, without the need for extra storage. One can e.g. first transmit a thumbnail image, and gradually transmit and decompress more data to increase the resolution and overall image quality.

- Wavelets can not only be used for image compression, but also for various image processing operations. The possibility to combine image processing and compression is a very appealing factor. However, image processing cannot be done on the final encoded data stream: it must be done before the wavelet coefficients are quantized or encoded. But even then we win because the wavelet transform is a common factor in both image processing and image compression.

4.5.4 Zerotree Encoding

Most compression methods are targeted at “small” images. In that case the individual subbands are too small to get a high compression gain from compressing them separately. Furthermore, even after the decorrelating step of a wavelet transform, there is still correlation present among spatially related wavelet coefficients in subbands at different resolution levels in the multi-resolution decomposition. That’s why compression algorithms based on wavelets often use a variant of zerotree encoding [42, 39].

Zerotree encoding combines the subbands to exploit the remaining correlation. If a coefficient at a low-resolution level is small (or even zero), chances are high

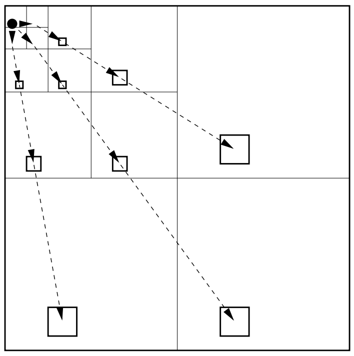

Figure 4.7: Zerotree encoding: a tree of coefficients at different resolution levels that correspond to one low-pass coefficient.

that the corresponding coefficients at higher resolution levels are also small (or zero). In that case the whole tree of coefficients that offsprings from one low-pass coefficient can be encoded in an efficient way (see Figure 4.7).

Zerotree encoding permits the construction of an embedded coder. Such a coder produces a bit stream in which the bits are generated in order of importance, so it can be truncated wherever desired, to meet specific bandwidth or storage requirements, while providing the best reconstruction quality for that size of the truncated bit stream. This is a form of progressive transmission: first the most important parts of the image are transmitted, while later parts of the bit stream contain the necessary information to gradually increase the image resolution and overall quality.

4.6 Conclusions

Many image processing operations like noise reduction and image scaling can be done on wavelet transformed images. Furthermore wavelets can also be used for image compression, giving better performance (higher signal-to-noise ratios, better visual quality and higher compression rates) than other compression methods. This combination of image processing and compression is an important feature of wavelet-based processing.