Compressing Still and Moving Images with Wavelets [*]

Michael L. Hilton   Björn D. Jawerth   Ayan Sengupta

April 18, 1994[†]

Abstract

The wavelet transform has become a cutting-edge technology in image compression research. This article explains what wavelets are and provides a practical, nuts-and-bolts tutorial on wavelet-based compression that will help readers to understand and experiment with this important new technology.

Keywords: image coding, signal compression, wavelet transform, image transforms

1 Introduction

The advent of multimedia computing has lead to an increased demand for digital images. The storage and manipulation of these images in their raw form is very expensive; for example, a standard 35mm photograph digitized at 12 μm per pixel requires about 18 MBytes of storage and one second of NTSC-quality color video requires almost 23 MBytes of storage. To make widespread use of digital imagery practical, some form of data compression must be used.

Digital images can be compressed by eliminating redundant information. There are three types of redundancy that can be exploited by image compression systems:

- *Spatial Redundancy.* In almost all natural images, the values of neighboring pixels are strongly correlated.
- *Spectral Redundancy.* In images composed of more than one spectral band, the spectral values for the same pixel location are often correlated.
- *Temporal Redundancy.* Adjacent frames in a video sequence often show very little change.

The removal of spatial and spectral redundancies is often accomplished by transform coding, which uses some reversible linear transform to the decorrelate the image data (Rabbani and Jones 1991). Temporal redundancy is exploited by techniques that only encode the differences between adjacent frames in the image sequence, such as motion prediction and compensation (Jain and Jain 1981; Liu and Zaccarin 1993).

In the last few years, the wavelet transform has become a cutting edge technology in image compression research. Although the literature on wavelets is vast, most of the papers

[*]The work in this paper was supported by Summus, Ltd.

[†]**To appear in** *Multimedia Systems*, **Vol. 2, No. 3**

dealing with wavelet-based image compression are written by specialists for the specialist. The purpose of this article is to provide a practical, nuts-and-bolts tutorial on wavelet-based compression that will (hopefully) help you to understand and experiment with this important new technology.

This paper is organized into three main sections. Section 2 discusses the theory behind wavelets and why they are useful for image compression. Section 3 describes how the wavelet transform is implemented and used in still image compression systems, and presents some results comparing several different wavelet coding schemes with the JPEG (Wallace 1991) still image compression standard. In Section 4 we describe some initial results with a novel software-only video decompression scheme for the PC environment. We conclude the paper with some remarks about current and future trends in wavelet-based compression.

1.1 A Note on Performance Measures

Throughout this paper, numbers are given for two measures of compression performance — compression ratio and peak signal-to-noise ratio (PSNR). The results of both of these performance measures can be used to mislead the unwary reader, so it is important to explain exactly how these figures were computed. We define compression ratio as

$$\frac{\text{the number of bits in the original image}}{\text{the number of bits in the compressed image}}.$$

In this paper we confine our measurements to 8 bits per pixel (bpp) greyscale images, so the peak signal-to-noise ratio in decibels (dB) is computed as

$$\text{PSNR} = 20 \log_{10} \frac{255}{\text{RMSE}}$$

where RMSE is the root mean-squared error defined as

$$\text{RMSE} = \sqrt{\frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} [f(i,j) - \hat{f}(i,j)]^2}$$

and N and M are the width and height, respectively, of the images in pixels, f is the original image, and $\hat{f}$ is the reconstructed image. Note that the original and the reconstructed images must be the same size.

2 Wavelets

The purpose of this section is to provide an intuitive understanding of what wavelets are and why they are useful for signal compression. For a more rigorous introduction to wavelets, see (Daubechies 1992), (Chui 1992), or (Jawerth and Sweldens 1992).

One of the most commonly used approaches for analyzing a signal $f(x)$ is to represent it as a weighted sum of simple building blocks, called *basis functions*:

$$f(x) = \sum_i c_i \Psi_i(x)$$

where the $\Psi_i(x)$ are basis functions and the c_i are coefficients, or weights. Since the basis functions Ψ_i are fixed, it is the coefficients which contain the information about the signal.

The simplest such representation uses translates of the impulse function as its only bases, yielding a representation that reveals information only about the time domain behavior of the signal. Choosing the sinusoids as the basis functions yields a Fourier representation that reveals information only about the signal's frequency domain behavior.

For the purposes of signal compression, neither of the above representations is ideal. What we would like to have is a representation which contains information about both the time and frequency behavior of the signal. More specifically, we want to know the frequency content of the signal at a particular instant in time. However, resolution in time (Δx) and resolution in frequency ($\Delta \omega$) cannot both be made arbitrarily small at the same time because their product is lower bounded by the Heisenberg inequality

$$\Delta x \Delta \omega \geq \frac{1}{2}.$$

This inequality means that we must trade off time resolution for frequency resolution, or vice versa. Thus, it is possible to get very good resolution in time if you are willing to settle for low resolution in frequency, and you can get very good resolution in frequency if you are willing to settle for low resolution in time.

The situation is really not all that bad from a compression standpoint. By their very nature, low frequency events are spread out (or non-local) in time and high frequency events are concentrated (or localized) in time. Thus, one way that we can live within the confines of the Heisenberg inequality and yet still get useful time-frequency information about a signal is if we design our basis functions to act like cascaded octave bandpass filters, which repeatedly split the signal's bandwidth in half.

To gain insight into designing a set of basis functions that will satisfy both our desire for information and the Heisenberg inequality, let us compare the impulse function and the sinusoids. The impulse function cannot provide information about the frequency behavior of a signal because its support — the interval over which it is non-zero — is infinitesimally small. At the opposite extreme are the sinusoids, which cannot provide information about the time behavior of a signal because they have infinite support. What we seek, then, is a compromise between these two extremes: a set of basis functions $\{\Psi_i\}$, each with finite support of a different width. The different support widths allow us to trade off time and frequency resolution in different ways; for example, a wide basis function can examine a large region of the signal and resolve low frequency details accurately, while a short basis function can examine a small region of the signal to resolve time details accurately.

To simplify things, let us constrain all of the basis functions in $\{\Psi_i\}$ to be scaled and translated versions of the same prototype function Ψ, known as the *mother wavelet*. The scaling is accomplished by multiplying x by some scale factor; if we choose the scale factor to be a power of 2, yielding $\Psi(2^\nu x)$ where ν is some integer, we get the cascaded octave bandpass filter structure we desire. Because Ψ has finite support, it will need to be translated along the time axis in order to cover an entire signal. This translation is accomplished by considering all the integral shifts of Ψ,

$$\Psi\left(2^\nu x - k\right), \qquad k \in \mathcal{Z}.$$

Note that this really means that we are translating Ψ in steps of size $2^{-\nu}k$.[1] Putting this all together gives us a *wavelet decomposition* of the signal,

$$f(x) = \sum_{\nu \text{ finite}} \sum_{k \text{ finite}} c_{\nu k} \Psi_{\nu k}(x)$$

[1]This is because $\Psi(2^\nu x - k) = \Psi(2^\nu(x - 2^{-\nu}k))$.

where

$$\Psi_{\nu k}(x) = 2^{\nu/2}\Psi\left(2^{\nu}x - k\right)$$

(the multiplication by $2^{\nu/2}$ is needed to make the bases orthonormal). So far we have said nothing about the coefficients $c_{\nu k}$. They are computed by the *wavelet transform*, which is just the inner product of the signal $f(x)$ with the basis functions $\Psi_{\nu k}(x)$.

The comparisons between wavelets and octave bandpass filters was not made just for pedagogical reasons. Wavelets can, in fact, be thought of and implemented as octave bandpass filters, and we shall treat them as such for the remainder of this paper.

3 Still Image Compression

A wide variety of wavelet-based image compression schemes have been reported in the literature, ranging from simple entropy coding to more complex techniques such as vector quantization (Antonini et al. 1992; Hopper and Preston 1992), adaptive transforms (Desarte et al. 1992; Wickerhouser 1992), tree encodings (Shapiro 1993; Lewis and Knowles 1992), and edge-based coding (Froment and Mallat 1992). All of these schemes can be described in terms of the general framework depicted in Fig. 1. Compression is accomplished by applying a wavelet transform to decorrelate the image data, quantizing the resulting transform coefficients, and coding the quantized values. Image reconstruction is accomplished by inverting the compression operations. We now describe each of the boxes in Fig. 1 in more detail.

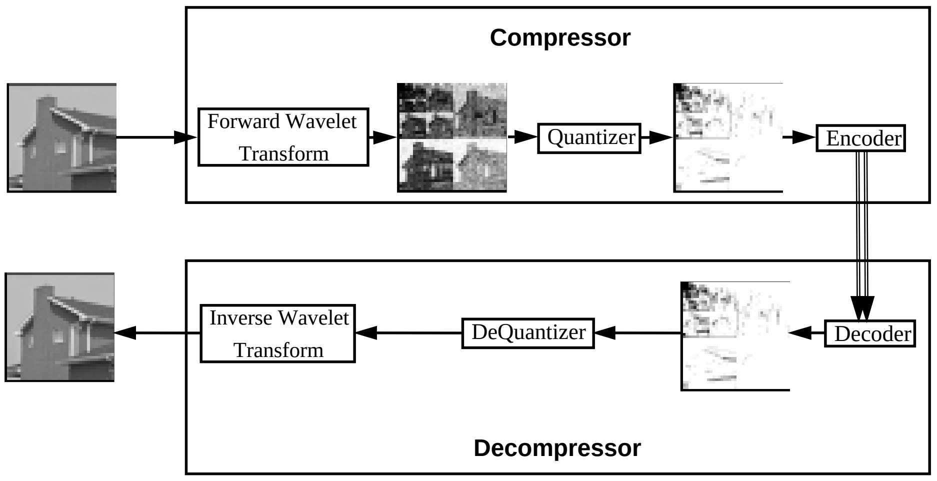

Figure 1: Block diagram of wavelet-based image coders.

3.1 Implementing the Wavelet Transform

The forward and inverse wavelet transforms can each be efficiently implemented in $\mathcal{O}(n)$ time by a pair of appropriately designed Quadrature Mirror Filters (QMFs) (Croisier et al. 1976). Therefore, wavelet-based image compression can be viewed as a form of subband coding (Woods and O'Neil 1986). Each QMF pair consists of a lowpass filter (H) and a highpass filter (G) which split a signal's bandwidth in half. The impulse responses of H and G are mirror images, and are related by

$$g_n = (-1)^{1-n} h_{1-n}. \tag{1}$$

The impulse responses of the forward and inverse transform QMFs — denoted $(\tilde{H}, \tilde{G})$ and (H, G) respectively — are related by

$$\begin{aligned} g_n &= \tilde{g}_{-n} && (2)\\ h_n &= \tilde{h}_{-n}. && (3) \end{aligned}$$

To illustrate how the wavelet transform is implemented, we shall use Daubechie's $\mathcal{W}_6$ wavelet (Daubechies 1988). We chose this wavelet because it is well known and has some nice properties. One such property is that it has two vanishing moments, which means the transform coefficients will be zero (close to zero) for any signal that can be described by (approximated by) a polynomial of degree 2 or less. The mother wavelet basis for $\mathcal{W}_6$ is shown in Fig. 2. The filter coefficients for H of $\mathcal{W}_6$ are

$$\begin{aligned} h_0 &= 0.332670552950\\ h_1 &= 0.806891509311\\ h_2 &= 0.459877502118\\ h_3 &= -0.135011020010\\ h_4 &= -0.085441273882\\ h_5 &= 0.035226291882 \end{aligned}$$

from which the coefficients for G, $\tilde{H}$, and $\tilde{G}$ can be derived using Equations 1, 2, and 3. The impulse responses of H and G are shown in Fig. 3.

A one-dimensional signal s can be filtered by convolving the filter coefficients c_k with the signal values:

$$\hat{s}_i = \sum_{k=0}^{M} c_k s_{i-k}$$

where M is the number of coefficients, or *taps*, in the filter. The one-dimensional forward wavelet transform of a signal s is performed by convolving s with both $\tilde{H}$ and $\tilde{G}$ and downsampling by 2. As dictated by Equation 1, the relationship of the $\tilde{H}$ and $\tilde{G}$ filter coefficients with the beginning of signal s is

$$\begin{array}{ccccccccccc} & & & & \tilde{h}_5 & \tilde{h}_4 & \tilde{h}_3 & \tilde{h}_2 & \tilde{h}_1 & \tilde{h}_0 & & \\ & & & & s_0 & s_1 & s_2 & s_3 & s_4 & s_5 & s_6 & \cdots \\ \tilde{g}_5 & \tilde{g}_4 & \tilde{g}_3 & \tilde{g}_2 & \tilde{g}_1 & \tilde{g}_0 & & & & & & \end{array}$$

Note that the $\tilde{G}$ filter extends before the signal in time; if s is finite, the $\tilde{H}$ filter will extend beyond the end of the signal. A similar situation is encountered with the inverse wavelet transform filters H and G. In an implementation, one must make some choice about what

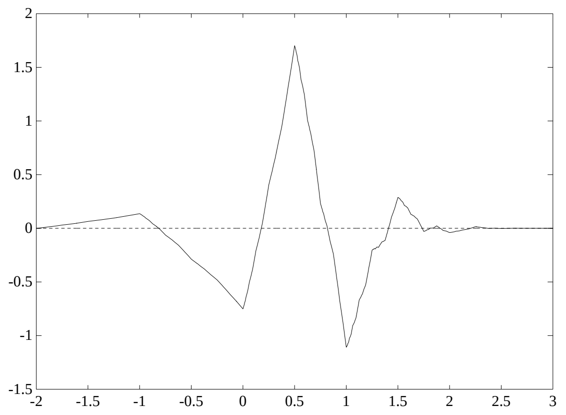

Figure 2: The mother wavelet basis function for $\mathcal{W}_6$.

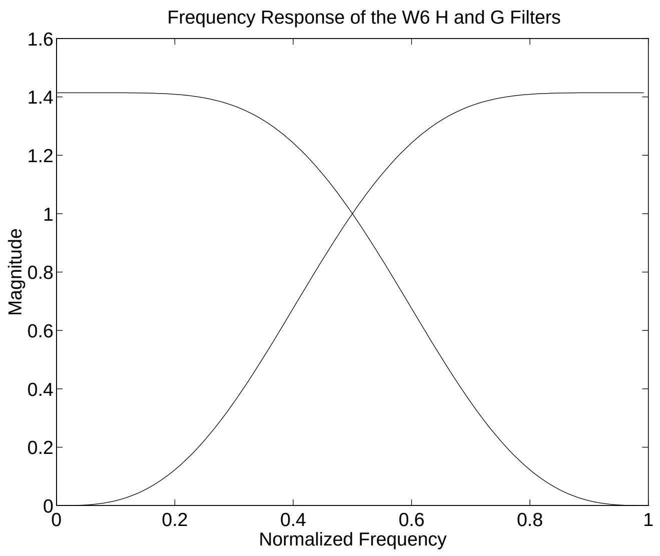

Figure 3: Frequency response of the $\mathcal{W}_6$ QMFs.

values to pad the extensions with. A choice which works well in practice is to wrap the signal about its endpoints, i.e.,

$$\cdots\ s_{n-1}\ s_n\ \boxed{s_0\ s_1\ s_2\ \cdots\ s_{n-2}\ s_{n-1}\ s_n}\ s_0\ s_1\ \cdots,$$

thereby creating a periodic extension of s.

Fig. 4 illustrates a single 2-D forward wavelet transform of an image, which is accomplished by two separate 1-D transforms. The image $f(x, y)$ is first filtered along the x dimension, resulting in a lowpass image $f_L(x, y)$ and a highpass image $f_H(x, y)$. Since the bandwidth of f_L and f_H along the x dimension is now half that of f, we can safely downsample each of the filtered images in the x dimension by 2 without loss of information. The downsampling is accomplished by dropping every other filtered value. Both f_L and f_H are then filtered along the y dimension, resulting in four subimages: f_{LL}, f_{LH}, f_{HL}, and f_{HH}. Once again, we can downsample the subimages by 2, this time along the y dimension. As illustrated in Fig. 4, the 2-D filtering decomposes an image into an *average signal* (f_{LL}) and three *detail signals* which are directionally sensitive: f_{LH} emphasizes the horizontal image features, f_{HL} the vertical features, and f_{HH} the diagonal features. The directional sensitivity of the detail signals is an artifact of the frequency ranges they contain.

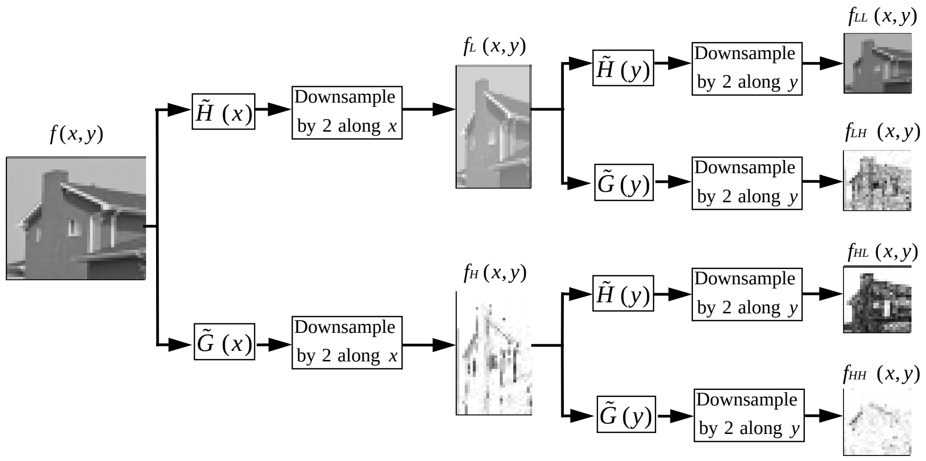

Figure 4: Block diagram of the 2-D forward wavelet transform.

It is customary in wavelet compression to recursively transform the average signal.[2] The number of transformations performed depends on several factors, including the amount of compression desired, the size of the original image, and the length of the QMF filters. In general, the higher the desired compression ratio, the more times the transform is performed.

After the forward wavelet transform is completed, we are left with a matrix of coefficients that comprise the average signal and the detail signals of each scale. No compression of the original image has been accomplished yet; in fact, each application of the forward wavelet

[2]A more sophisticated decomposition strategy is to use the *wavelet packets* of Coifman and Meyer (Wickerhouser 1992; Coifman and Wickerhouser 1992).

transform causes the magnitude of the coefficients to grow, so there has actually been an increase in the amount of storage required for the image! Compression is achieved by quantizing and encoding the wavelet coefficients.

The 2-D inverse wavelet transform is illustrated in Fig. 5. The average and detail signals are first upsampled by 2 along the y dimension. Upsampling is accomplished by inserting a zero between each pair of values in the y dimension. The upsampling is necessary to recover the proper bandwidth required to add the signals back together. After upsampling, the signals are filtered along the y dimension and added together appropriately. The process is then repeated in the x dimension, yielding the final reconstructed image.

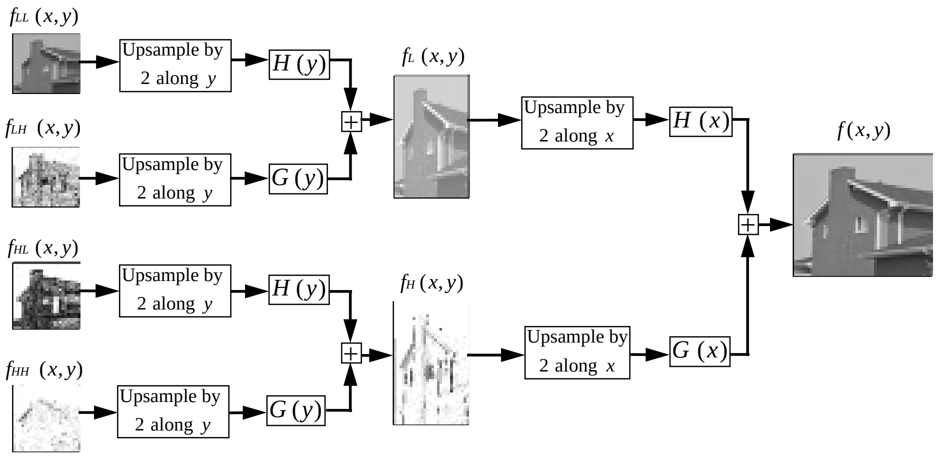

Figure 5: Block diagram of the 2-D inverse wavelet transform.

3.2 Quantization

The forward wavelet transform decorrelates the pixel values of the original image and concentrates the image information into a relatively small number of coefficients. Fig. 6 (Left) is a histogram of the pixel values for the 8-bits per pixel (bpp) 512×512 Lena image, and Fig. 6 (Right) is a histogram of the wavelet coefficients of the same image after the forward wavelet transform is applied. The "information packing" effect of the wavelet transform is readily apparent from the scarcity of coefficients with large magnitudes.

The sharply peaked coefficient distribution of the wavelet transformed image has a lower zero-th order entropy (4.24 bpp) than the original image (7.46 bpp), thereby increasing the amount of lossless compression possible.

We can also take advantage of the energy invariance property of the wavelet transform to achieve high-quality lossy compression. The energy invariance property says that total amount of energy in an image does not change when the wavelet transform is applied. This property can also be viewed in a slightly different way: any changes made to the values of the wavelet coefficients will result in proportional changes in the pixel values of the reconstructed image. In other words, we can eliminate (set to zero) those coefficients with small magnitudes without creating significant distortion in the reconstructed image. In

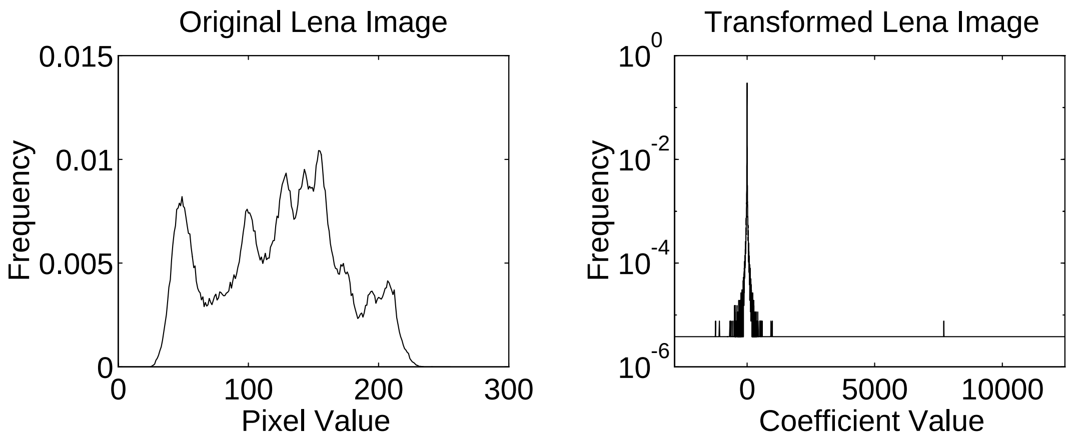

Figure 6: LEFT) Normalized histogram of the pixel values in the original Lena image. RIGHT) Normalized histogram of the wavelet transform coefficients of the same image.

practice, it is possible to eliminate all but a few percent of the wavelet coefficients and still get a reconstructed image of reasonable quality. The elimination of small valued coefficients can be accomplished by applying a *thresholding function*

$$T(t,x) = \begin{cases} 0 & \text{if } |x| < t \\ x & \text{otherwise} \end{cases}$$

to the coefficient matrix. The amount of compression obtained can now be controlled by varying the threshold parameter t.

Higher compression ratios can be obtained by *quantizing* the non-zero wavelet coefficients before they are encoded. A quantizer is a many-to-one function $Q(x)$ that maps many input values into a (usually much) smaller set of output values. Quantizers are staircase functions characterized by a set of numbers $\{d_i, i = 0, \ldots, N\}$ called *decision points* and a set of numbers $\{r_i, i = 0, \ldots, N-1\}$ called *reconstruction levels*. An input value x is mapped to a reconstruction level r_i if x lies in the interval $(d_i, d_{i+1}]$.

To achieve the best results, a separate quantizer should be designed for each scale, taking into account both the properties of the Human Visual System (Marr 1982) and the statistical properties of the scale's coefficients. The characteristics of the Human Visual System guide the allocation of bits among the different scales, and the coefficient statistics guide the quantizer design for each scale. Descriptions of various bit allocation strategies can be found in (Matic and Mosley 1993) and (Clarke 1985).

The distribution of coefficient values in the various detail signals can be modeled reasonably well by the Generalized Gaussian Distribution (GGD). The probability density function of the coefficient distribution at each scale ν, then, can be given by (Abramowitz and Stegun 1965):

$$p_\nu(x) = \left[\frac{\alpha_\nu \eta(\alpha_\nu, \sigma_\nu)}{2,\,(1/\alpha_\nu)}\right] \exp\left(-[\eta(\alpha_\nu, \sigma_\nu)\,|x|]^{\alpha_\nu}\right)$$

Scale ν	Codeword Size (in bits)	Decision Points and Reconstruction Levels	
8	2	d_i	5, 10, 23, 48, 256
		r_i	7, 15, 32, 65
7	3	d_i	5, 10, 18, 28, 40, 57, 81, 117, 512
		r_i	7, 14, 22, 33, 47, 67, 95, 139
6	3	d_i	10, 16, 26, 41, 63, 95, 144, 223, 1024
		r_i	12, 20, 32, 50, 76, 114, 174, 271
5	5	d_i	20, 33, 51, 73, 99, 128, 161, 201, 245 291, 339, 386, 436, 500, 591, 738, 2048
		r_i	25, 41, 61, 85, 113, 143, 179, 223, 267 314, 362, 410, 461, 539, 644, 834

Table 1: Lloyd-Max quantizers generated using magnitude data from the $\mathcal{W}_6$ transformed Lena image.

where

$$\eta(\alpha, \sigma) = \sigma^{-1}\left[\frac{,\,(3/\alpha)}{,\,(1/\alpha)}\right]^{1/2}$$

and

$$,\,(a) = \int_0^\infty e^{-t} t^{a-1} dt.$$

σ_ν is the standard deviation of the coefficient distribution at scale ν and α_ν is a shape parameter describing the exponential rate of decay of the distribution at scale ν. For example, when $\alpha_\nu = 1$ the GGD becomes the Laplacian pdf, while $\alpha_\nu = 2$ leads to the Gaussian pdf. The α_ν appropriate for a particular class of images can be computed using the χ^2 test or by simple observation. For the $\mathcal{W}_6$ wavelet and the Lena image, several of the appropriate values of α_ν and σ_ν are:

$$\begin{array}{rclcrcl} \alpha_8 & = & 0.58 & \quad & \sigma_8 & = & 5.48 \\ \alpha_7 & = & 0.49 & & \sigma_7 & = & 9.39 \\ \alpha_6 & = & 0.43 & & \sigma_6 & = & 14.33 \\ \alpha_5 & = & 0.39 & & \sigma_5 & = & 20.17 \end{array}$$

The design of scalar quantizers will also depend on the type of encoder to be used. If the encoder uses fixed-length codewords, the Lloyd-Max algorithm (Max 1960) can be used to design a quantizer that minimizes the mean-squared quantization error. If a variable-length entropy coder is used, uniform quantization is optimal (in the mean-squared error sense) when the coefficient distribution is Laplacian; for other distributions, the algorithm in (Wood 1969) can be used to design an optimal quantizer. Vector quantization (Gersho and Grey 1992) has also been used in wavelet compression systems, for example (Antonini et al. 1992) and (Bradley and Brislawn 1993).

Table 1 lists the decision points and reconstruction levels for a set of Lloyd-Max quantizers generated using magnitude data from the $\mathcal{W}_6$ transformed Lena image. One extra bit per codeword is needed to represent the sign of the quantized coefficient. The codeword sizes were chosen by experimentation.

3.3 Coding the Coefficients

The encoder/decoder pair, or *codec*, has the task of losslessly compressing and decompressing the sparse matrix of quantized coefficients. Codec design has received a tremendous amount of attention, and a wide variety of schemes exist (Lelewer and Hirshberg 1987). The design of a codec is usually a compromise between (often conflicting) requirements for memory use, execution speed, available bandwidth, and reconstructed image quality.

For applications requiring fast execution, simple run-length coding (Pratt 1978) of the zero-valued coefficients has proven very effective. (The distribution of non-zero coefficients is such that rarely is it profitable to run-length encode them.) The zero run-lengths can be encoded using either fixed-length codewords or variable-length entropy coding; entropy coding is more expensive to implement, but can improve the peak signal to noise ratio (PSNR) of reconstructed images by as much as 3 dB, depending upon the energy-packing ability of the wavelet in use.

For applications requiring the best possible image quality at a particular compression ratio, a technique such as Shapiro's Zero Tree encoding (Shapiro 1993) is a better choice. The execution speed tradeoff between these two codecs is quite dramatic: our run-length entropy coder takes less than one second to compress a 512×512 8 bpp image on a 66-MHz 80486 computer, and Zero Tree-like coders can take up to 45 seconds to compress the same image on the same machine. However, the quality of the Zero Tree image is much better — 36.28 dB PSNR (Shapiro 1993) vs. 33.2 dB PSNR at a compression ratio of 16:1.

3.4 Compression Results

The peak signal to noise ratios of several different wavelet compression techniques applied to the 512×512 8-bpp Lena image are compared in Fig. 7. The graphs show that both the encoding technique and the particular wavelet used can make a significant difference in the performance of a compression system: the Zerotree coder performs the best; biorthogonal wavelets (Antonini et al. 1992; Cohen 1992; Averbuch et al. 1993) perform better than $\mathcal{W}_6$; and variable length coders perform better than fixed length coders.

The performance of a baseline JPEG (Wallace 1991) image compressor[3] is also indicated in Fig. 7. At compression ratios less than 25:1 or so, JPEG performs better numerically than the simple wavelet coders. At compression ratios above 30:1, JPEG performance rapidly deteriorates, while wavelet coders degrade gracefully well beyond ratios of 100:1. Figure 8 compares the visual quality of several image coders.

4 Video Compression

The wavelet transform can also be used in the compression of image sequences, or video. Video compression techniques are able to achieve high quality image reconstruction at low bit rates by exploiting the temporal redundancies present in an image sequence (Le Gall 1991; Liu 1991). Wavelet-based implementations of at least two standard video compression techniques, hierarchical motion compensation (Uz et al. 1991) and 3-D subband coding (Karlson and Vetterli 1989), have been reported (Zhang and Zafar 1992; Lewis and Knowles 1990). However, the computational expense of the wavelet transform has so far prevented its use in realtime, software-only video codecs for PC-class computers. In this section, we

[3]The JPEG coder that is included in Version 2.21 of John Bradley's `xview` program was used to generate the JPEG performance data shown in Fig. 7.

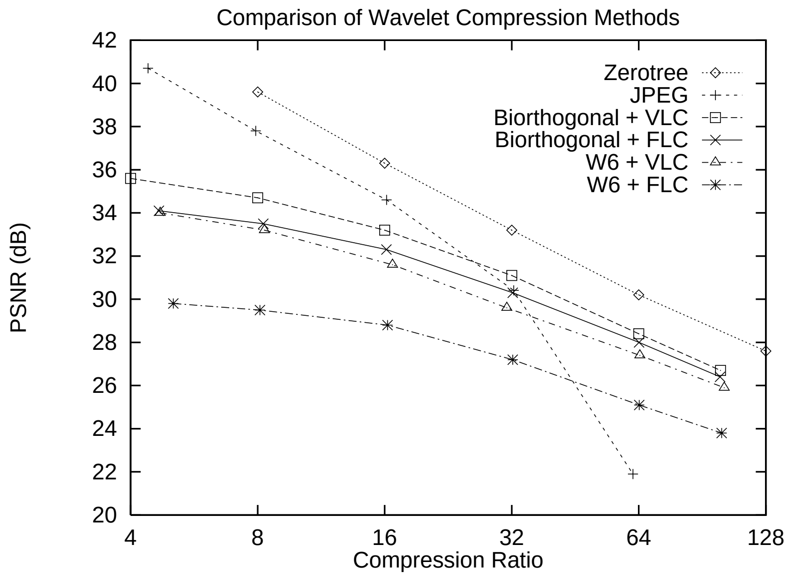

Figure 7: A comparison of the image reconstruction quality of several different wavelet coders and JPEG. The tests were performed on the 512×512, 8-bpp Lena image. "VLC" means Variable Length Coder, and "FLC" means Fixed Length Coder.

describe a new technique for rapidly evaluating the inverse wavelet transform and illustrate its use in the context of a software-only video decoder based on frame differencing.

4.1 The Basic Idea

The playback speed of a wavelet-based video coder depends in large part upon how long it takes to perform the inverse wavelet transform. A 66-Mhz 80486 computer takes about 0.25 seconds to compute a complete inverse wavelet transform for a 256×256, 8-bpp greyscale image. Unless one finds a way to avoid performing a complete inverse transform each time an image frame is reconstructed, wavelets are not viable for software-only video of reasonably sized images.

Fortunately, it is not necessary to perform the complete inverse transform for each frame in a slowly varying image sequence. The value of an arbitrary pixel p in an image is determined by a weighted sum of all the basis vectors in the wavelet decomposition that include p in their region of support. If the weights (i.e., the wavelet coefficients) of these basis vectors do not change between frames in an image sequence, then the value of pixel p will not change either. Therefore, it is not necessary to compute the inverse wavelet transform for those regions of the image that have not changed between frames. This idea was first put forth in (Andersson et al. to appear).

The basic idea for rapidly decompressing image sequences, then, is to only compute the inverse wavelet transform for those pixels influenced by coefficients that have change by a meaningful amount between adjacent frames in the sequence.

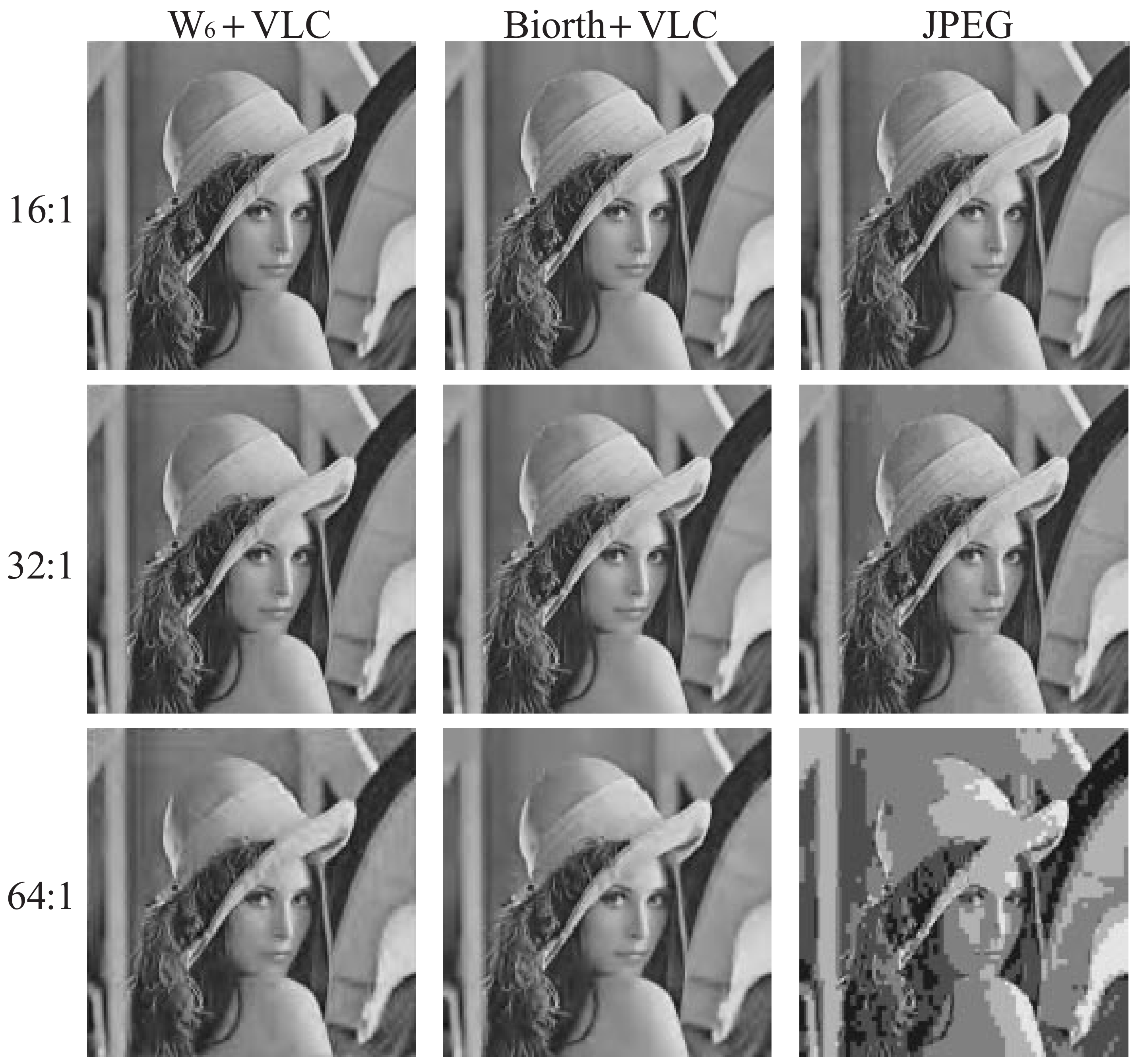

Figure 8 - Reconstructed images for the W6+VLCT Biorthogonal+VLCT and JPEG image coders.

4.2 A Simple Frame Differencing Video Coder

The simple video coder described herein is shown in Figs. 9 and 10. Let us consider a sequence of images $\{f_i\}_{i=0,1,\ldots}$, where each f_i denotes the ith frame in the sequence. The difference between two adjacent frames is given by

$$\Delta f_i = f_{i+1} - f_i.$$

Δf_i is called a *difference image*, and it contains only the change in image content between frame f_i and f_{i+1} — it does not contain any redundant first-order temporal information. There is spatial redundancy in Δf_i, however, and this redundancy can be reduced by application of some wavelet transform W.[4] Thresholding can now be performed on the transformed difference image $W(\Delta f_i)$ to eliminate image changes that are considered too small to be meaningful. After thresholding, we now have an approximate transformed difference image $\widehat{W(\Delta f_i)}$ that is extremely sparse. $\widehat{W(\Delta f_i)}$ is then analyzed to determine which portions of the inverse wavelet transform will need to be performed to reconstruct an approximation $\widehat{\Delta f_i}$ of the ith difference image. This information is then encoded and sent to the video decoder.

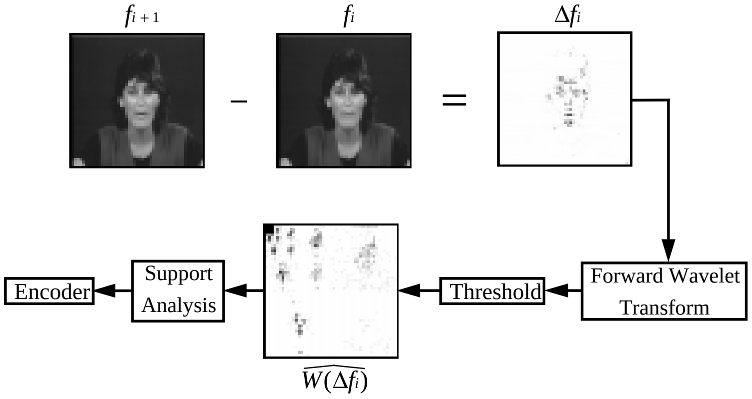

Figure 9: Block diagram of the video encoder.

Using the information sent by the encoder, the decoder can reconstruct $\widehat{\Delta f_i}$. Because of its sparse nature, $\widehat{\Delta f_i}$ can be reconstructed very quickly by computing the inverse wavelet transform for only those pixels influenced by the coefficients sent by the encoder. We assume that the decoder has available some approximation $\hat{f}_i$ of frame i, so the next frame in the sequence can be constructed as

$$\hat{f}_{i+1} = \hat{f}_i + \widehat{\Delta f_i}.$$

We have implemented a prototype video compression system based on the ideas described above and achieved promising initial results. The results of an experiment in which we compressed 30 frames of the standard Miss America video sequence (the images were first rescaled to 256×256 pixels) are presented in Table 2. The experiment was performed on a 66 MHz 80486 computer running the OS/2 operating system, and the entire video

[4]We note that because the wavelet transform is linear, it does not matter from a theoretical standpoint if we form $W(\Delta f_i)$ by $W(f_{i+1}) - W(f_i)$ or by $W(f_{i+1} - fi)$.

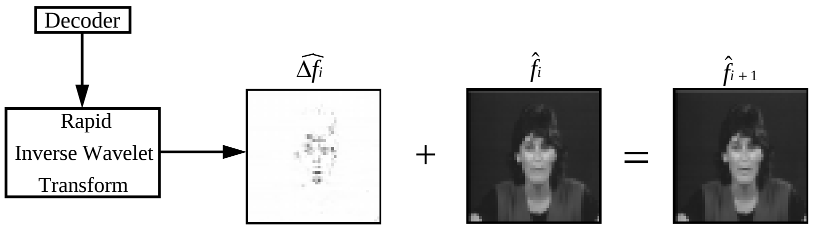

Figure 10: Block diagram of the video decoder.

Threshold	Compressed Size (Bytes)	Compression Ratio	Transform (sec)	Decompression (sec)	Speed (fps)	PSNR (dB)
0	172,2278	11:1	0.074	0.157	6.4	40.73
10	150,977	13:1	0.073	0.150	6.6	38.03
20	107,351	18:1	0.061	0.121	8.3	35.06
30	91,365	22:1	0.056	0.115	8.6	32.22

Table 2: Results of Compression experiments on 30 frames of the Miss America video sequence. The original, uncompressed sequence requires 1,966,080 bytes of storage. All times and PSNRs are mean values for the entire sequence.

coder and decoder are written in the C programming language. The wavelet we use is an integerized version of Daubechies' $\mathcal{W}_6$.

Performing the complete 2-dimensional inverse wavelet transform takes 0.25 seconds per frame. Our experiments indicate that at compression ratios of around 20:1, the partial 2-dimensional inverse transform technique is more than four times faster than the full inverse transform, and is capable of transforming over 16 image frames per second. This measurement is only of the time required to perform the inverse wavelet transform — it does not include the time it takes to decode the image data or display the reconstructed image.

Our codec currently uses a combination of fixed and variable length codes for representing the video data. Our primary concern so far has been increasing the speed of the inverse wavelet transform, and we have not paid much attention to coding issues. The development of codes which can be quickly decoded is of major importance, because the time required for decoding the compressed image data is presently the performance bottleneck of our experimental decompression system.

5 Concluding Remarks

Basic and applied research in the field of wavelets has made tremendous progress in the last five years. Image compression schemes based on wavelets are rapidly gaining maturity, and have already begun to appear in commercial software/hardware systems. The reconstruction quality of wavelet compressed images has already moved well beyond capabilities of JPEG, which is the current international standard for image compression.

Video is the next big challenge for wavelet-based data compression. Our laboratory experiments using three dimensional wavelet transforms to compress a $64 \times 64 \times 64$ color video sequence indicate that visually lossless video compression is possible at compression rates near 1000:1, but the memory and processor requirements are presently too great to make such a scheme practical. The technique presented in Section 4 of this paper is a small step towards a practical video compression scheme, but much research remains to be done.

It is also interesting to note that wavelet research on image compression has had a strong impact on several areas of numerical analysis, especially in the solution of partial differential equations (Alpert 1992; Alpert et al. 1993; Beylkin et al. 1991). The compression of an image, which is just a matrix of intensity values, is not really different from compressing the kernel matrix of a functional operator. The compressed operator is a sparse matrix, and sparse matrix operations can often be performed orders of magnitude faster than their non-sparse counterparts. Undoubtedly, this will lead to new results in numerical analysis that will impact image compression, leading to better algorithms in areas such as computer vision.

References

Abramowitz M, Stegun IA (1965) Handbook of Mathematical Functions. Dover, New York.

Alpert B, Beylkin G, Coifman R, Rokhlin V (1993) Wavelet-like bases for the fast solution of second-kind integral equations. SIAM Journal of Scientific Computing, 14(1):159–184.

Alpert B (1992) Wavelets and other bases for fast numerical linear algebra. In: Chui CK (ed) Wavelets: A Tutorial in Theory and Applications, Volume 2. Academic Press, San Diego, pp. 181–216.

Andersson L, Hall N, Jawerth B, Peters G (1994) Wavelets on closed subsets of the real line. In: Schumacher LL, Webb G (ed) Recent Advances in Wavelet Analysis. Academic Press, San Diego, pp. 1–61.

Antonini M, Barlaud M, Mathieu P, Daubechies I (1992) Image coding using wavelet transform. IEEE Trans Image Processing, 1(2):205–220.

Averbuch A, Lazar D, Israeli M (1993) Image compression using wavelet transform and multiresolution decomposition. In Storer JA, Cohn M (ed) Proceedings Data Compression Conference 93. IEEE Computer Society Press, p. 459.

Beylkin G, Coifman R, Rokhlin V (1991) Fast wavelet transforms and numerical algorithms I. Commun Pure and Applied Math, 44:141–183.

Bradley JN, Brislawn CM (1993) Wavelet transform-vector quantizer compression of supercomputer ocean models. In: Storer JA, Cohn M (ed) Proceedings Data Compression Conference 93. IEEE Computer Society Press, pp. 224–233.

Chui CK (1992) An Introduction to Wavelets, volume 1 of Wavelet Analysis and Its Applications. Academic Press, San Diego.

Clarke RJ (1985) Transform Coding of Images. Academic Press, San Diego.

Cohen A (1992) Biorthogonal wavelets. In: Chui CK (ed) Wavelets: A Tutorial in Theory and Applications, Volume 2. Academic Press, San Diego, pp. 123–152.

Coifman R, Wickerhauser MV (1992) Entropy-based algorithms for best basis selection. IEEE Trans Information Theory, 38(2):713–718, Part II.

Croisier A, Esteban D, Galand C (1976) Perfect channel splitting by use of interpolation/decimation tree decomposition techniques. In: Int Conf on Information Science and Systems.

Daubechies I (1988) Orthonormal bases of compactly supported wavelets. Commun Pure and Applied Math, 41:909–996.

Daubechies I (1992) Ten Lectures on Wavelets, volume 61 of CBMS-NSF Regional Conference Series in Applied Mathematics. SIAM.

Desarte P, Macq B, Slock DTM (1992) Signal-adapted multiresolution transform for image coding. IEEE Trans Information Theory, 38(2):897–904.

Froment J, Mallat S (1992) Second generation compact image coding with wavelets. In: Chui CK (ed) Wavelets: A Tutorial in Theory and Applications, Volume 2. Academic Press, San Diego, pp. 655–678.

Le Gall D (1991) MPEG: A video compression standard for multimedia applications. Commun ACM, 34(4):46–58.

Gersho A, Gray RM (1992) Vector Quantization and Signal Compression. Kluwer Academic Publishers, Boston Dordrecht London.

Hopper T, Preston F (1992) Compression of grey-scale fingerprint images. In: Storer JA, Cohn M (ed) Proceedings Data Compression Conference 92. IEEE Computer Society Press, pp. 309–318.

Jain JR, Jain AK (1981) Displacement measurement and its application in interframe image coding. IEEE Trans Communications, COM-29(12):1799–1808.

Jawerth B, Sweldens W (1992) An overview of wavelet based multiresolution analyses.

Technical report, Industrial Mathematics Initiative, The University of South Carolina Department of Mathematics.

Karlsson G, Vetterli M (1989) Packet video and its integration into the network architecture. IEEE J Selected Areas in Communications, 7(5):739–751.

Lelewer DA, Hirshberg DS (1987) Data compression. ACM Computing Surveys, 19(3):261–295.

Lewis AS, Knowles G (1990) Video compression using 3D wavelet transforms. Electronics Letters, 26(6):396–397.

Lewis AS, Knowles G (1992) Image compression using the 2-d wavelet transform. IEEE Trans Image Processing, 1(2):244–250.

Liu B, Zaccarin A (1993) New fast algorithms for the estimation of block motion vectors. IEEE Trans Circuits Systems for Video Technology, 3(2):148–157.

Liu M (1991) Overview of the px64 kbit/s video coding standard. Commun ACM, 34(4):59–63.

Marr D (1982) Vision: A Computational Investigation into the Human Representation and Processing of Visual Information. W. H. Freeman.

Matic R, Mosley J (1993) Wavelet transform-adaptive scalar quantization of multispectral data. In: AIAA Computing in Aerospace 9 Conference. American Institute of Aeronautics and Astronautics, addendum 93–4485.

Max J (1960) Quantizing for minimum distortion. IRE Trans on Information Theory, IT-6(1):7–12.

Pratt WK (1978) Digital Image Processing. Wiley, New York.

Rabbani M, Jones P (1991) Digital Image Compression Techniques, volume TT7 of SPIE Tutorial Texts in Optical Engineering. SPIE Press.

Shapiro JM (1993) Embedded image coding using zerotrees of wavelet coefficients. IEEE Trans Signal Processing, 41(12):3445–3462.

Uz KM, Vetterli M, Le Gall D (1991) Interpolative multiresolution coding of advanced television with compatible subchannels. IEEE Trans Circuits Systems for Video Technology, 1(1):86–99.

Wallace GK (1991) The JPEG still picture compression standard. Commun ACM, 34(4):30–44.

Wickerhauser MV (1992) Acoustic signal compression with wavelet packets. In: Chui CK (ed) Wavelets: A Tutorial in Theory and Applications, Volume 2. Academic Press, San Diego, pp. 679–700.

Wood RC (1969) On optimum quantization. IEEE Trans Information Theory, IT-15(2):248–252.

Woods JW, O'Neil SD (1986) Subband coding of images. IEEE Trans Acoustics, Speech, Signal Proc, ASSP-34(5):1278–1288.

Zhang Y, Zafar S (1992) Motion-compensated wavelet transform coding for color video compression. IEEE Trans Circuits Systems for Video Technology, 2(3):285–296.