WAVELET IMAGE COMPRESSOR MINIMAGE

Hao Gu, Don Hong*, and Martin Barrett
Department of Computer and Information Sciences
Department of Mathematics*
East Tennessee State University
Johnson City, Tennessee

August 31, 2000

Abstract

Nowadays, still images are used everywhere in the digital world. The shortages of storage capacity and transmission bandwidth make efficient compression solutions essential. A revolutionary mathematics tool, wavelet transform, has already shown its power in image processing.

MinImage, the major topic of this paper, is an application that compresses still images by wavelets. MinImage is used to compress grayscale images and true color images. It implements the wavelet transform to code standard BMP image files to LET wavelet image files, which is defined in MinImage. The code is written in C++ on the Microsoft Windows NT platform.

This paper illustrates the design and implementation details in MinImage according to the image compression stages. First, the preprocessor generates the wavelet transform blocks. Second, the basic wavelet decomposition is applied to transform the image data to the wavelet coefficients. The discrete wavelet transforms are the kernel component of MinImage and are discussed in detail. The different wavelet transforms can be plugged in to extend the functionality of MinImage. The third step is the quantization. The standard scalar quantization algorithm and the optimized quantization algorithm, as well as the dequantization, are described. The last part of MinImage is the entropy-coding schema. The reordering of the coefficients based on the Peano Curve and the different entropy coding methods are discussed. This paper also gives the specification of the wavelet compression parameters adjusted by the end user.

The interface, parameter specification, and analysis of MinImage are shown in the final appendix.

1 Introduction

There are many image transmissions through the Internet around the world everyday. The widespread, consumer-market use of information in the form of images has contributed much to the development of data compression techniques. Large amounts of data can create enormous problems in both storage and transmission. For example, a single A4 (8.27" width and 11.69" height) size color picture, scanned at 300 dpi with 24 bits/pixel of true color, will produce approximately 25 Megabytes of data without being compressed. At least 17 floppy disks are required to store such a picture. This picture requires more than 6 minutes for transmission by a 64k bit/s ISDN line. In a distributed environment, large image files remain a major bottleneck within such system. One possible solution is to increase the bandwidth, but the relatively high cost makes this a less attractive solution. Therefore, compression is a necessary and essential method for creating image files with the manageable and transmittable sizes. This paper will focus on a still image compressor (MinImage) design and its implementation based on some basic wavelet transforms.

A digital image is represented by a matrix of numeric values each representing a quantized intensity value. The points at which an image is sampled are called pixels. At each pixel location, the brightness and the chrominance of the image are sampled and quantized. The pixel values in intensity images are called gray scale levels or colors. True color images are multi-spectral images when the spectral sampling is restricted to three bands, and these correspond to the red, green, and blue wavelengths to which the human visual system responds.

The design goal of image compression is to represent images with as few bits as possible, according to some fidelity criteria, to save both storage and transmission channel capacities. The basic principle of image compression techniques is getting rid of the inherent redundancies.

There are two basic categories of image processing techniques. One is lossless encoding, which is reversible and does not sacrifice any information. The other technique is lossy encoding, which causes image quality degradation in the compression step. Careful consideration of human visual perception can optimize the algorithm to make the degradation less recognizable, though it depends on the selected compression ratio. Wavelet analysis is a new mathematical tool, which can be regarded as an extension of Fourier analysis. It has already shown its arguably revolutionary impacts upon a wide range of applications.

The visual system of a human being works in the same way as a wavelet transform. The cells of visual system react to both frequency and space in the same manner as wavelet transform, which means that a wavelet transform can encode natural scenes concisely. By wavelet transform, the original values of a two-dimensional image data can be packed into a relatively small number of large magnitude coefficients. The wavelet coefficients only indicate changes, areas with no change or very small change give small or zero coefficients. One important property of the wavelet transform is energy invariance: The total amount of energy in the image does not change after the wavelet transform is performed. It implies that any change in the wavelet coefficients is proportional to the change in the reconstructed image. Therefore, the low magnitude coefficients can be ignored without significantly distorting the reconstructed image. By truncating or removing these small coefficients from the representation introduces only small errors in the reconstructed image. Thus, the sparse data coding makes wavelets an excellent lossy solution to image compression.

The wavelet image compressor, MinImage, is designed for compressing either 24-bit true color or 8-bit gray scale digital images. It is a lossy compressor, which means that the decompressed image is not quite the same as the one before compression.

MinImage is designed to exploit known limitations of the human eyes. The wavelets are intended for compressing images that will be looked at by humans. If the image is to be analyzed by machines, it may be not proper to use the wavelets to compress the images, because the small errors introduced by the wavelets may cause big problems in the occurrence of machine based recognition, even if these errors are invisible to the human eye.

A very useful property of MinImage is that the degree of compression, as well as the quality of the image, can be varied by adjusting the compression parameters through the interface of the program. One major advantage of this compression engine is that the user can trade off between the compressed image file size and the image quality. A very small image file size can be achieved if relatively poor image quality is acceptable, such as in applications like indexing image archives. Conversely, the user can also enhance the image quality by allowing lesser compression. The flexibility of fine tuning the compression parameters is another goal, as it makes MinImage a very useful tool to teach and study wavelets applications in image compression.

MinImage is also one of the state of the art practical wavelets applications in image compression. Wavelets are likely to be the basis of the next generation of image compression standards, but they are perhaps ten years behind JPEG in the standardization pipeline. The follwoing diagram (Figure 1) gives the baseline schema of MinImage.

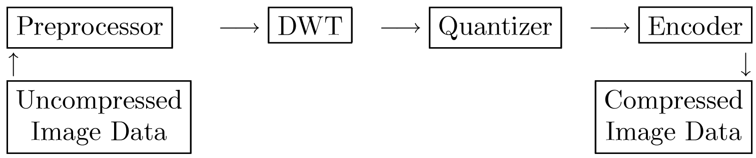


Figure 1. The Baseline Schema of MinImage

2 Preprocess

The main purpose of the preprocess is to generate the wavelet transform blocks from the raw image data. One possible optimization, in order to achieve better compression ratio, is to downsample (resample an input signal at a lower rate) the true color image data. Figure 2 and 3 describe the preprocess schemas in both compression and uncompression stages.

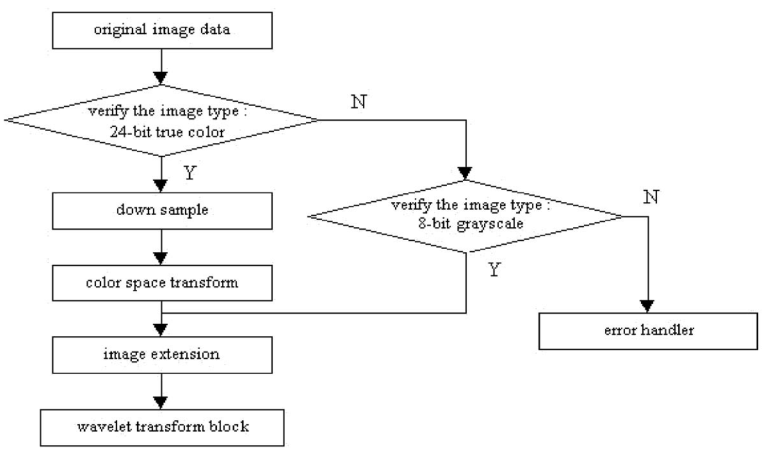


Figure 2. The Preprocess In The Compression

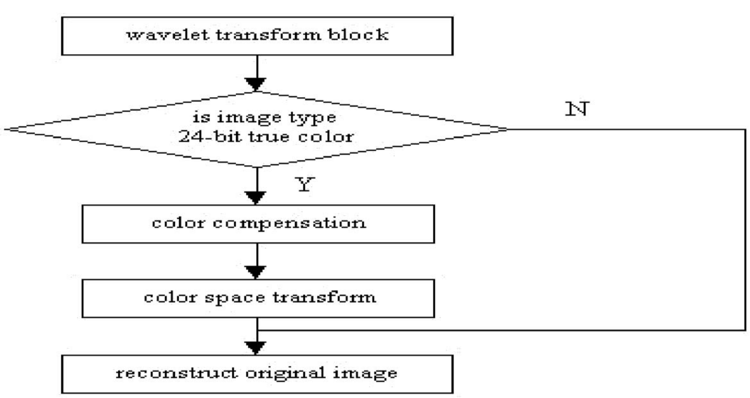


Figure 3. The Preprocess In The Uncompression Stage

2.1 Color Spaces Transform

MinImage takes advantage of the fact that small color changes are perceived less accurately than small changes in brightness. The human eyes are much more sensitive to brightness variations than to hue variations. Therefore, the hue data can be compressed more heavily thsn the brightness data [1]. RGB is a way that a computer defines a color in terms of the extensions of red, green, and blue components on CRT. YC_bC_r color space (see [10]) was used internally in the MinImage. The following are the RGB to YC_bC_r and YC_bC_r to RGB conversion equations.

$$\begin{aligned} Y &= 0.2989R + 0.5866G + 0.1145B \\ C_b &= -0.1688R - 0.3312G + 0.5B \\ C_r &= 0.5R - 0.4184G - 0.0816B \end{aligned} \tag{1}$$

$$\begin{aligned} R &= Y + 1.4022C_r \\ G &= Y - 0.3456C_b - 0.7145C_r \\ B &= Y + 1.7710C_b \end{aligned} \tag{2}$$

The color space transform is necessary for compressing the true color images since the Y space contains more information than the C_b and the C_r spaces. Thus, the compressor can treat these color spaces differently in order to get greater compression ratio. By comparing the three color spaces, we conclude that the C_b and C_r color spaces are easier to be compressed than the Y color space. The C_b and C_r color spaces contain less important information.

Table 1. The Down Sample Modes

Sample Mode	Description	Total Compression Ratio
H1V1	In the Y, C_b and C_r color spaces, the ratio of the original image resolution to the sampled image resolution is $1:1$, both horizontally and vertically. No downsample is performed.	1:1
H2V1	Downsample is only applied in the Cb and Cr spaces horizontally. In the Cb and Cr color spaces, the original image horizontal resolution: the sampled image horizontal resolution $= 2:1$.	2:3
H2V2	Downsample is only applied in the Cb and Cr spaces both horizontally and vertically. In the Cb and Cr color spaces, the original image resolution: the sampled image resolution $= 2:1$, both horizontally and vertically.	1:2

2.2 Resolution Reductions

When dealing with true color images, in order to reduce the image source data, downsampling is implemented in the Cb and Cr color spaces. There are three different sampling schemas: H1V1, H2V1 and H2V2, which were studied by Steinmetz and Nahrstedt in [10]. They are described in Table 1. The total compression ratio in Table 1 is defined as the original image data size to the compressed image data size.

2.3 Image Extensions

Image extension is the last step in the image preprocessing. After the image data are extended, the data can be divided into blocks, which are ready to be wavelet transformed. The advantage of using wavelet transform blocks is to make the wavelet transform process independent of the source data context, so that the wavelet transform code can be reused in other applications.

The wavelet transform block is defined as a square data block with the width of a power of two. The two dimensions are required by the two-dimensional wavelet transforms. Even if a one-dimensional wavelet transform is used, this data block can be regarded as a one-dimensional stream of data. However, in the image compression applications, a two-dimensional wavelets transform is preferred because the data in an image are not only horizontally correlated but also vertically correlated. The square shape of a block is required to implement exactly the same wavelet transforms on both dimensions. The advantage is the efficient and easy implementation of the two-dimensional wavelet transforms based on the symmetry in both dimensions. The disadvantage is the extensions of source image data are necessary because of the random original image sizes that MinImage supports. In the worst case, the compression ratio is reduced as the wavelet transform block is defined as a very large square, but the original image is a narrow line. However, the defect is not so significant, because the user can always adjust the wavelet transform block size in order to make it match the image size as well as possible.

Given the size of the wavelet transform block, the whole image is divided into the blocks by doing extensions on the boundary blocks if needed. See Figure 4.

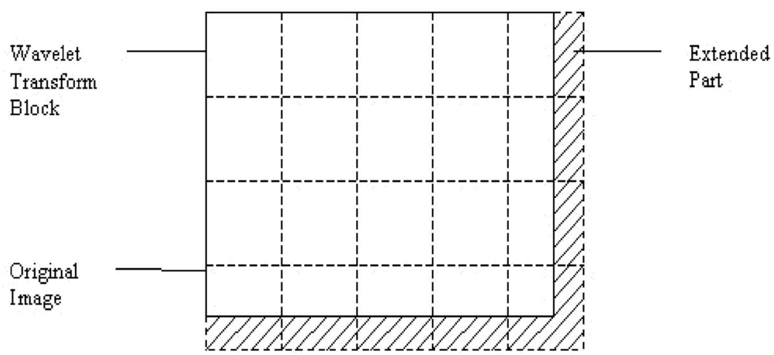


Figure 4. Image Divisions And Extensions

With the flexibility in assigning the size of the wavelet transform block, a special case occurs when an image can be defined in just one block, if the width and the height of the image do not exceed the domain of the block sizes, which can be adjusted by the end user.

The image extension algorithm is shown below. The following is a boundary wavelet transform block, which needs to be extended. The block is divided into four areas. The shaded area is defined as area S, which has all the sampled pixel values in the original image. The white area A, B, and C have all the extended pixel values. See Figure 5.

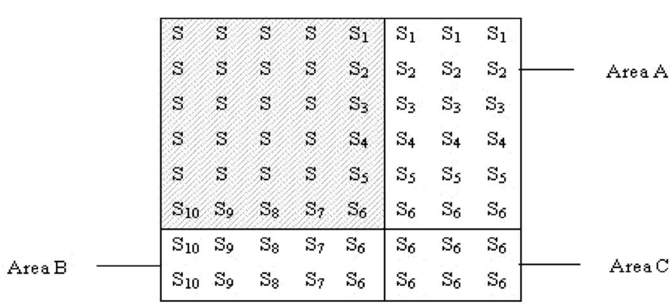


Figure 5. Boundary Wavelet Transform Block

The image extension algorithm is defined as follows. Figure 6 describes the data structures in MinImage used to represent the wavelet transform blocks.

PROCEDURE Image extension in the boundary wavelet transform block
IF the pixel is in area A
    The value of pixel = the value of the right most sampled pixel
ELSE IF the pixel is in area B
    The value of pixel = the value of the top most sampled pixel
ELSE IF the pixel is in area C
    The value of pixel = the value of the right bottom sampled pixel
END IF
END PROCEDURE

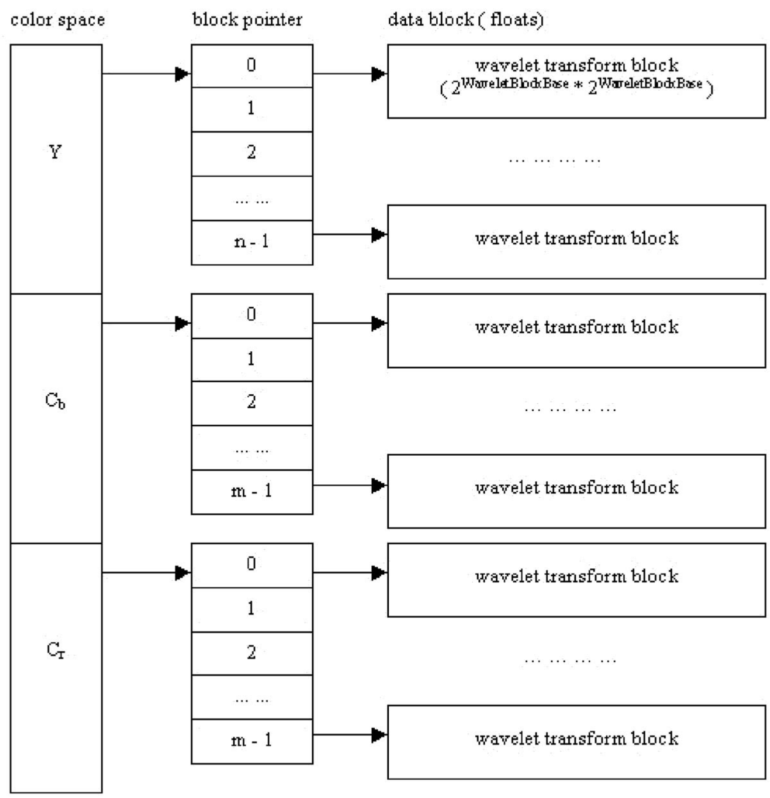


Figure 6. The Data Structure Of The Wavelet Transform Blocks

3 DISCRETE WAVELET TRANSFORM

The wavelet transform is a projection of a signal onto a series of basis functions called the wavelet basis. The DWT (Discrete Wavelet Transform) is the kernel part of MinImage. It is applied to the wavelet transform blocks generated by the preprocessor.

The simplest form of wavelets, the Haar basis, is used as an option in MinImage and is discussed here to demonstrate the application of wavelets in image compression. The discussion will first focus on how a one-dimensional function can be decomposed by using the Haar wavelets and their application to decompose and compose a one-dimensional signal. Then, the two- dimensional Haar basis functions and the two dimensional wavelet decomposition and composition algorithms will be explored. Finally, the coefficient reduction method is represented as a real compression stage in the wavelet image compressor. In this paper, the wavelet transform is also called the wavelet decomposition. The inverse wavelet transform equals wavelet composition. We use the following notation:

(1) ϕ is defined as the scaling function. $\phi_i = \phi(x-i), i \in Z$ expand the vector space V and $\phi_i^j = \phi(2^j x - i), i \in Z$ expand the vector space V^j.

(2) ψ is defined as the wavelet function. $\psi_i = \psi(x-i), i \in Z$ expand the vector space W and $\psi_i^j = \phi(2^j x - i), i \in Z$ expand the vector space W^j.

The Haar scaling function $\phi = \phi_0^0$ is defined as

$$\phi_0^0(x) = \begin{cases} 1, & 0 \le x \le 1; \\ 0, & \text{otherwise.} \end{cases}$$

The Haar wavelet function $\psi = \psi_0^0$ is defined as

$$\psi_0^0(x) = \begin{cases} 1, & 0 \le x \le \frac{1}{2}; \\ -1, & \frac{1}{2} \le x \le 1; \\ 0, & \text{otherwise.} \end{cases}$$

The scaling functions satisfy the following dilation equation

$$\phi(x) = \sum_{k=0}^{n} c_k \phi(2x-k),$$

where $[0, n]$ is the support of the scaling function ϕ. For Haar scaling function, $c_0 = c_1 = 1$ and $c_k = 0$ for $k \neq 0, 1$.

Usually, the wavelet function is generated by the scaling function as following:

$$\psi_0^0(x) = \sum_{k=0}^{n} (-1)^k c_{1-k} \phi_0^0(2x-k).$$

For the Haar wavelet, we have

$$\psi(x) = \psi(2x) - \psi(2x-1).$$

Notice that the vector spaces V^j, $j = 0, 1, \cdots, n$ are nested:

$$V^0 \subset V^1 \subset V^2 \subset \cdots \subset V^n.$$

Let W^j be the orthogonal complement of the V^j in the space V^{j+1}. Then W^j can be used to represent the parts of functions in V^{j+1} that cannot be represented in the space V^j. Using the direct sum notation, we have

$$V^n = V^{n-1} \oplus W^{n-1} = V^{n-2} \oplus W^{n-2} \oplus W^{n-1} = \cdots = V^0 \oplus W^0 \oplus W^1 \oplus \cdots \oplus W^{n-1}.$$

3.1 The 1D Haar Wavelets Decomposition and Composition

An operator L is defined to perform the smoothing operations. By operator L, the original signal is smoothed and scaled by $\frac{1}{2}$. The resolution of the new signal is reduced. In Haar wavelet transform, the operator L:

$$V^n \xrightarrow{L} V^{n-1}$$

is defined as $L = \text{diag}[[1/2\ 1/2], \cdots, [1/2\ 1/2]]$.

While the operator L calculates the averaged signal in the specified level, operator H computes the wavelet amplitude, resulting in the wavelet coefficients at that level. The new signal given by the operator H is the difference between the original signal and the expanded version of the new smoothed signal from the L operation. The resolution of the result from the operator H is also reduced by 1/2 in Haar wavelet transform:

$$V^n \overset{H}{\to} W^{n-1},$$

where $H = \text{diag}[[1/2 \ \ -1/2], \cdots, [1/2 \ \ -1/2]]$.

3.1.1 1-D Haar Wavelet Decomposition

The operators L and H comprise the wavelet decomposition. By rolling together the matrix operations of L and H, a single interleaved matrix can be defined to compute the wavelet decomposition. For wavelets with two nonzero coefficients c_0 and c_1, the wavelet decomposition matrix W is defined as following:

$$W = \text{diag}[C, C, \cdots, C],$$

where $C = \begin{pmatrix} c_0 & c_1 \\ c_1 & -c_0 \end{pmatrix}$. The wavelet decomposition can be computed by

$$\begin{pmatrix} s_0 & d_0 & s_1 & d_1 & \cdots\cdots & s_{\frac{n}{2}-1} & d_{\frac{n}{2}-1} \end{pmatrix}^t = W \begin{pmatrix} x_0 & x_1 & \cdots\cdots & x_{n-1} \end{pmatrix}^t,$$

where A^t is the transpose matrix of A.

In the Haar wavelet decomposition, the wavelet coefficients are $c_0 = 1$ and $c_1 = 1$. In order to simplify the wavelet composition (the inverse wavelet transform), the transform matrix is adjusted by a scale factor of $\frac{1}{\sqrt{2}}$, so that the normalized Haar wavelet coefficients are $c_0 = \frac{1}{\sqrt{2}}$ and $\frac{1}{\sqrt{2}}$.

3.1.2 Haar Wavelet Composition

Because $c_0^2 + c_1^2 = 1$, the decomposition matrix is orthogonal. The inverse matrix is the same as the transpose. Therefore, the wavelet composition matrix is $W^{-1} = W^t$.

The wavelet composition can be computed by

$$\begin{pmatrix} x_0 & x_1 & \cdots\cdots & x_{n-1} \end{pmatrix}^t = W_{n\times n}^{-1} \begin{pmatrix} s_0 & d_0 & s_1 & d_1 & \cdots\cdots & s_{\frac{n}{2}-1} & d_{\frac{n}{2}-1} \end{pmatrix}^t.$$

The one-dimensional wavelet composition can be described as Figure 7.

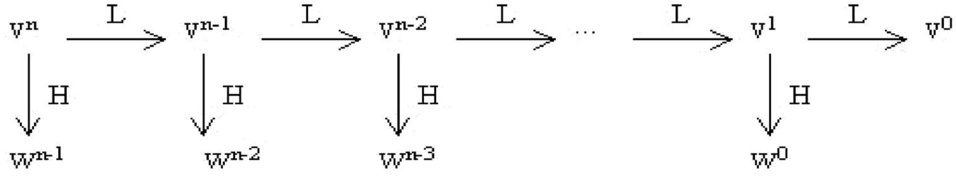


Figure 7. THE 1-D WAVELET COMPOSITION

3.2 1-D Haar Wavelet Decomposition and Composition Algorithms

Figure 8 is the pyramid wavelet decomposition algorithm (see [9] for example).

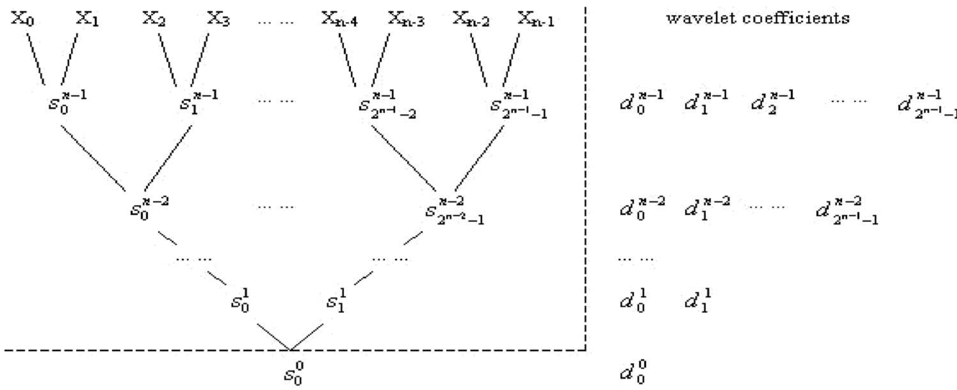


Figure 8. The Pyramid Wavelet Decomposition Algorithm

The 1-D Haar wavelet decomposition algorithm is represented in Figure 9. The *nMaxDecompositionStep* is defined to control when the wavelet decomposition should be stopped. For a complete wavelet decomposition of the signal X^n with 2^n entries, the *nMaxDecompositionStep* can be defined in the domain of $[1, n]$. *DecompositionStep* implements one step of the wavelet decomposition in an array of data. The 1-D Haar wavelet composition algorithm is defined as Figure 10. The *nStart* determines the start point of the wavelet composition.

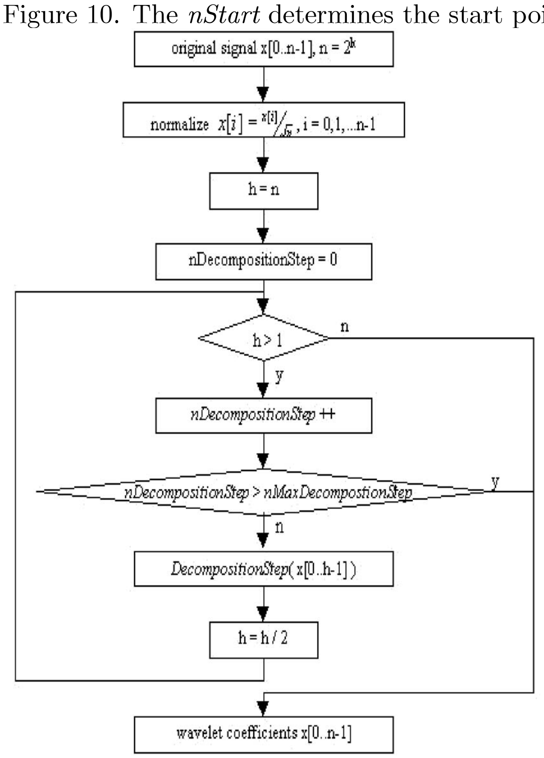


Figure 9. The 1-D Haar Wavelet Decomposition Algorithm In Minimage

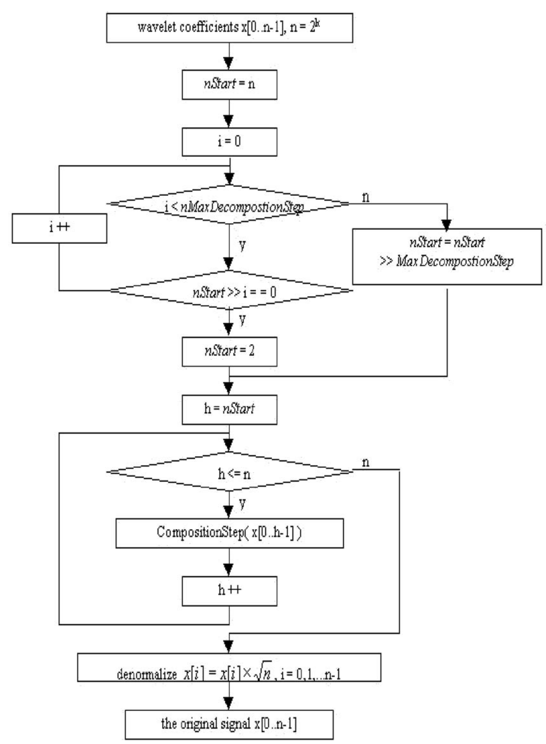


Figure 10. The 1-D Haar Wavelet Composition Algorithm In Minimage

3.3 The 2D Haar Wavelets Decomposition and Composition

There are two different methods of decomposing a two-dimensional image. The two methods yield coefficients that correspond to two different sets of basis functions. The standard decomposition of an image gives coefficients for a basis formed by the standard construction of a two-dimensional basis. The nonstandard decomposition gives coefficients for the nonstandard construction of basis functions. The standard construction of the basis consists of all the possible tensor products of one-dimensional basis functions. The nonstandard construction of a two dimensional basis proceeds by first defining a two dimensional scaling function.

3.3.1 The 2D Standard Wavelet Decomposition

In order to implement the standard decomposition of an image, the one-dimensional wavelet decomposition is applied first to each row of the pixel values. This operation gives an average value along with the detail coefficients for each row. Next, these transformed rows are treated as if they were themselves an image and the one-dimensional transform is applied to each column. The resulting values are all detail coefficients except for a single overall average coefficient.

PROCEDURE StandardDecomposition(ImageData[$0..n-1, 0..n-1$])
FOR row = 0 TO n-1 DO
    OneDimensionalDecomposition(ImageData[row, $0..n-1$])
END FOR
FOR col = 0 TO $n-1$ DO
OneDimensionalDecomposition(ImageData[row, $0..n-1$])
END FOR
END PROCEDURE

Figure 11 illustrates each step of the standard two-dimensional decomposition. The shaded parts are wavelet coefficients.

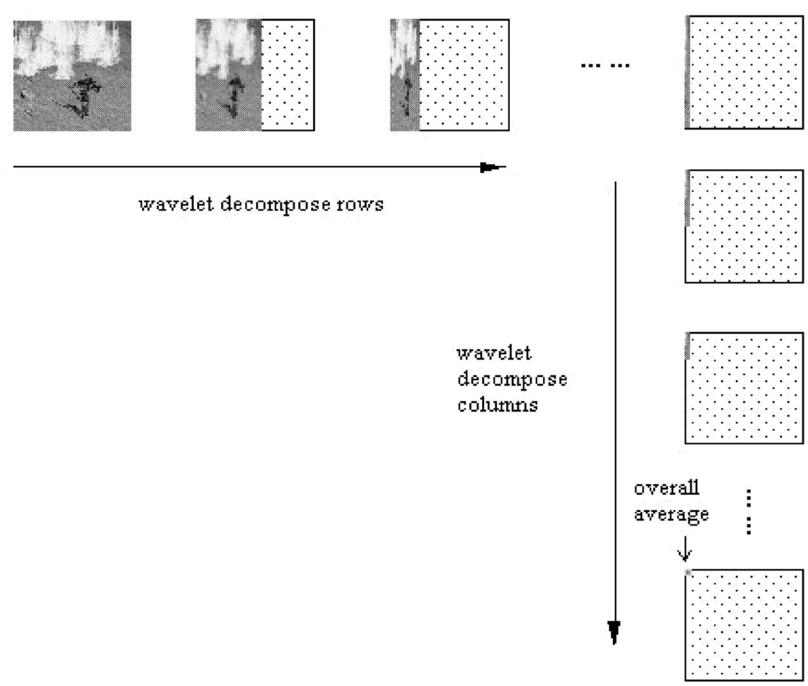


Figure 11. The 2d Standard Wavelet Decomposition Of An Image

3.3.2 The 2D Nonstandard Wavelet Decomposition

The nonstandard wavelet decomposition alternates between operations on the rows and columns. First, one step of horizontal wavelet decomposition is performed on the pixel values in each row of the image. Next, one step of the vertical wavelet decomposition is performed to each column of the previous result. The process is repeated to perform the complete wavelet decomposition.

PROCUDURE NonStandardDecomposition(ImageData[$0..n-1, 0..n-1$])
FOR row = 0 TO $n-1$ DO
FOR col = 0 TO $n-1$ DO
ImageData[row, col] $/= n$ $h = n$
WHILE $h > 1$ DO
FOR row = 0 TO $h-1$ DO
DecompositionStep(ImageData[row, $0..h-1$])
END FOR
FOR col = 0 TO $h-1$ DO
DecompositionStep(ImageData[$0..h-1, col$])
END FOR
$h = h/2$
END WHILE
END FOR
END FOR
END PROCEDURE

Figure 12 illustrates each step of the nonstandard two-dimensional decomposition. The shaded parts are wavelet coefficients.

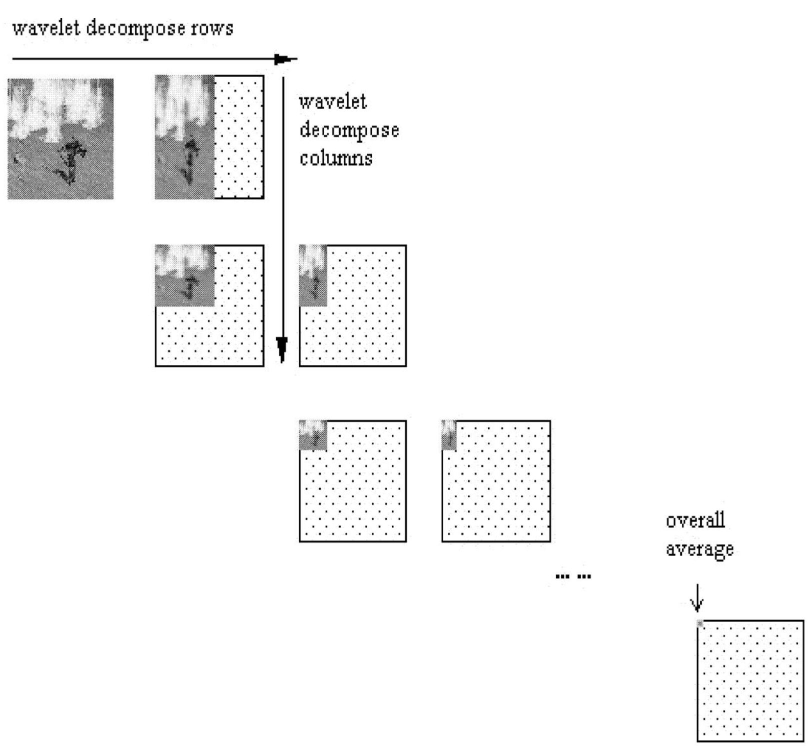


Figure 12. The 2D Nonstandard Wavelet Decomposition Of An Image

3.4 Other Wavelet Transforms

The 1D wavelet transform and 2D wavelet transform algorithms described in the previous sections are applied not only to the Haar wavelet transform but also to the other wavelet transforms, such as Daubechies. The only difference exists in the 1D wavelet transform. In the Haar transform, only two signals are involved in the calculations of smooth and difference values. In the DAUB4, for example, four signals should be calculated because of the number of wavelet coefficients in the DAUB4 is four. In the same way, six signals should be calculated together in the DAUB6, and eight signals are needed in one wavelet transform step in the DAUB8.

The following matrix (3) illustrates the wavelet transform matrix for the DAUB4. The transform matrix (in this case DAUB4) is applied to the input $(x_1, \cdots, x_{16})$ yielding smooth data $(s_1, \cdots, s_8)$ interleaved with detail data $(d_1, \cdots, d_8)$. The results are permuted to separate the smooth and detail data. The detail data are simply stored while the transform matrix is applied to the smoothed data. The final output is the same number of data points as the input and could be used to reconstruct the original data. Note the pattern of coefficients used for the diagonal of the transform matrix in the matrix (3). The rows consist of identical pairs of coefficients. The odd rows are smoothing filters while even rows yield the detail data. The values wrap in the last two rows, if DAUB4 is used because it is assumed that data represent a repeating waveform.

$$W = \left[\begin{array}{cccccccccccccccc}
c_0 & c_1 & c_2 & c_3 \\
c_3 & -c_2 & c_1 & -c_0 \\
 & & c_0 & c_1 & c_2 & c_3 \\
 & & c_3 & -c_2 & c_1 & -c_0 \\
 & & & & c_0 & c_1 & c_2 & c_3 \\
 & & & & c_3 & -c_2 & c_1 & -c_0 \\
 & & & & & & c_0 & c_1 & c_2 & c_3 \\
 & & & & & & c_3 & -c_2 & c_1 & -c_0 \\
 & & & & & & & & c_0 & c_1 & c_2 & c_3 \\
 & & & & & & & & c_3 & -c_2 & c_1 & -c_0 \\
 & & & & & & & & & & c_0 & c_1 & c_2 & c_3 \\
 & & & & & & & & & & c_3 & -c_2 & c_1 & -c_0 \\
 & & & & & & & & & & & & c_0 & c_1 & c_2 & c_3 \\
 & & & & & & & & & & & & c_3 & -c_2 & c_1 & -c_0 \\
 & & & & & & & & & & & & & & c_0 & c_1 \\
 & & & & & & & & & & & & & & c_3 & -c_2 \\
c_2 & c_3 \\
c_1 & -c_0
\end{array}\right] \quad (3)$$

The wavelet transform matrix in matrix (3) is a generic one. The Haar wavelet transform can be considered a special case with the two wavelet transform coefficients. Therefore, by implementing a more generic 1D wavelet transform algorithm, MinImage can handle not only Haar, but also DAUB4, DAUB6, DAUB8 and other wavelet transforms.

3.5 Wavelet Coefficients Reductions

The goal of the compression is to express an initial set of data using some smaller set of data, probably with some loss of information. The compression can be applied by ignoring the wavelet coefficients below some magnitude threshold. The percentage of the coefficients needed to be kept of each color spaces can be specified by the user in MinImage. This threshold can be calculated by the following formula:

threshold = the kth element of the coefficients in the current color space, which is sorted by the absolute values. k = (1-fKeepPercent) $\times$ total number of coefficients in the current color space.

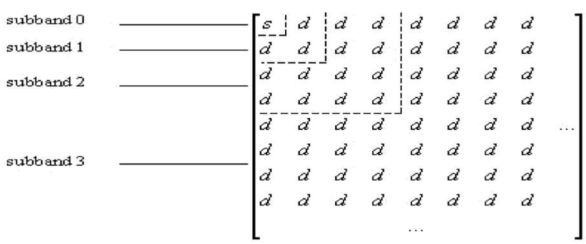
The coefficient reduction algorithm is illustrated as follows:

PROCEDURE ReduceCoefficients(float threshold)
  FOR EACH coefficients in the current color space DO
    IF | coefficient | < threshold

coefficient = 0
END IF
END FOR
END PROCEDURE

4 Wavelet Coefficients Quantizations

After the two-dimensional wavelet decomposition, the wavelet transform blocks contain the wavelet coefficients, which are real numbers. The task of the quantization stage is to quantize these coefficients to integers by using designated bits. The length used to represent each coefficient can be defined from one bit to eight bits. There is a trade- off between the precision and the compression ratio. The more bits used in quantizing each coefficient, the more information can be kept. However it will result in the lower compression ratio. Therefore, the design of the quantizer is important in MinImage. A block of the two-dimensional wavelet coefficients can be divided into subbands as Figure 13.



Subband	Coefficients Numbers
0	1
1	3
2	12
3	48
...	...

Figure 13. Subbands In A Wavelet Transform Block After The 2-D Wavelet Decomposition

Therefore, if the number of coefficients in the block is $n \times n$, the number of subbands is $\log_2 n + 1$. Given the subband $k, k = 0, 1, ..., \log_2 n$, the number of coefficients in the subband k is defined as:

$$\begin{cases} 1, & k = 0 \\ 3 \times 4^{k-1}, & k = 1, 2, \cdots, \log_2 n \end{cases}$$

4.1 Quantization Precision

In MinImage, all the wavelet transform blocks are quantized together, but each subband is quantized separately. The QuantizePrecision is defined as the maximum quantized value of the signal, so the domain of the quantized value will be [0, QuantizePrecision]. The different quantization precision can be set by the user in the different subband and in the different color space.

4.2 Standard Uniform Scalar Quantization and Dequantization

A scalar quantizer Q approximates the source signal X by $X = Q(X)$, which takes its values over to finite set. Suppose that X takes its values in the domain [MinCoefficient, MaxCoefficient], which is defined in the whole real axis. The [MinCoefficient, MaxCoefficient] is decomposed in QuantizePrecision intervals $\{(Xk-1, Xk]\}, 1 \leq k \leq$ QuantizePrecision of the same size, with X_0 = MinCoefficient and $X_{\text{QuantizePrecision}}$ = MaxCoefficient, $X_k - X_{k-1} = \Delta$, for $1 \leq k \leq$ QuantizePrecision. The uniform quantizer approximates $\forall X \in [X_{k-1}, X_k)$ by $Q(X) = Y_k$. The average quadratic distortion for this uniform quantizer is $D = \frac{\Delta^2}{12}$, see [8] by Mallat (1998). It is independent of the probability density of the source signal values. Suppose the QuantizePrecision is defined by

$$\text{QuantizedValue} = (int)(\,(\text{OringinalValue}-\text{MinCoefficient})\times\frac{\text{QuantizePrecision}}{\text{MaxCoefficient}-\text{MinCoefficient}}+0.5).$$

For example, the QuantizePrecision = 8, Figure 14 is the standard uniform scalar quantization schema. The minimal and maximal coefficients of each subband in the specific color space should be stored after the quantization process in order to dequantize the coefficients later.

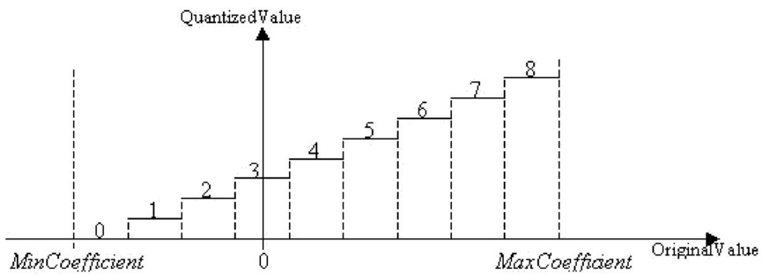


Figure 14. The Standard Uniform Scalar Quantization Method

4.3 The Optional Quantization Algorithm

The most important feature of the wavelet coefficients is the possible existance of a large number of zeros in some subbands, especially after the coefficient reductions. The disadvantage of the standard uniform scalar quantization is that the zeros in the different subbands are quantized to different values. The reason is that the *QuantizePrecision* in the different subbands may be different as defined by the user. These factors result in generating different quantized values for the same original value in the different subbands and color spaces, which has negative effect on the coefficients entropy coding.

The optimized quantization schema is applied to improve the compression ratio in the entropy coding by quantizing these coefficients according to their absolute value. The key point is to keep the original small value still small after being quantized. Therefore, at the same time, the number of zeros in the original coefficients will not be changed after being quantized. See Figure 15.

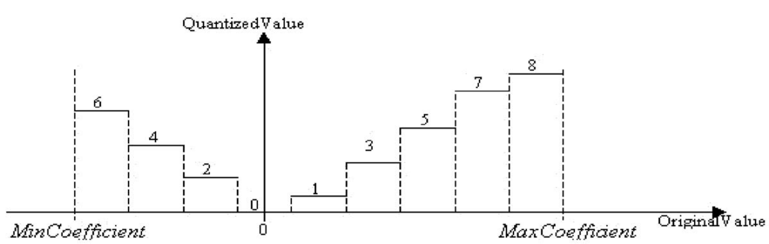


Figure 15. The Optimized Quantization Method

4.4 The Compression Ratio before the Entropy Encoding

The compression ratio can be calculated before the entropy encoding. This compression ratio is a theoretical value, because only the nonzero values are defined as the useful data. All the zeros are ignored in this case. There are several reasons for calculating the compression ratio at this stage. One reason is the entropy encoding is not the major research topic in this paper and the entropy-encoding program we will use was written by others. The second reason is the entropy-encoding actually influences the compression ratio significantly. Therefore, by just ignoring the zeros, a reasonable evaluation can be done on the wavelet transforms in the image compression applications, even though the practical compression ratio will be definitely reduced because of extra spaces storing those zero coefficients. See Appendix C for the compression ratio analysis in MinImage.

4.4.1 Grayscale Images

nwidth and *nHeight* are defined as the pixel width and the height of the original image. *nBlock* is defined as the width of the wavelet transform block. *WaveletTransformBlockNumbers* is defined as the number of the wavelet transform blocks in the whole image. *OriginalImageDataSize* is defined as the byte size of the original image data. It does not including the image header. *NewImageDataSize* is defined as the byte size of the compressed image data. It does not including the image header. The compression ratio can be calculated by the following equations:

$$\text{WaveletTransformBlockNumbers} = \lceil \frac{\text{nWidth}}{\text{nBlock}} \rceil \times \lceil \frac{\text{nHeight}}{\text{nBlock}} \rceil$$

$$\text{OriginalImageDataSize} = \text{nWidth} \times \text{nHeight}$$

$$\text{NewImageDataSize} = \log_2(\text{QuantizePrecision}[0][0] + 1) \times 1 +$$

$$+ \sum_{i=1}^{\log_2 nBlock} (\log_2(\text{QuantizePrecision}[0][i] + 1) \times 3 \times 4^{i-1})) \times \text{WaveletTransformBlockNumbers}$$

$$CompressionRatio = \frac{\text{NewImageDataSize}}{\text{OriginalImageDataSize}}.$$

4.4.2 True Color Images

nwidth and *nHeight* are defined as the pixel width and the height of the original image. *nBlock* is defined as the width of the wavelet transform block. *WaveletTransformBlockNumbersInY* is defined as the number of the wavelet transform blocks in the Y color spaces of the whole image. *WaveletTransformBlockNumbersInCrCb* is defined as the number of the wavelet transform blocks in the Cr and Cb color spaces of the whole image. *OriginalImageDataSize* is defined as the byte size of the original image data. It does not including the image header. *NewImageDataSize* is defined as the byte size of the compressed image data. It does not including the image header. The compression ratio can be calculated by the following equations:

$$\text{WaveletTransformBlockNumbers} = \lceil \frac{\text{nWidth}}{\text{nBlock}} \rceil \times \lceil \frac{\text{nHeight}}{\text{nBlock}} \rceil$$

$$\text{WaveletTransformBlockNumbersInCrCb} = \lceil \frac{\lceil \frac{\text{nWidth}}{\text{nHsample}} \rceil}{\text{nBlock}} \rceil \times \lceil \frac{\frac{\text{nHeight}}{\text{nVsample}}}{\text{nBlock}} \rceil \times 2$$

$$\text{NewImageDataSize} = (\log_2(\text{QuantizePrecision}[0][0] + 1) \times 1 +$$

$$+ \sum_{i=1}^{\log_2 nBlock} (\log_2(\text{QuantizePrecision}[0][i] + 1) \times 3 \times 4^{i-1})) \times$$

$$\times \text{WaveletTransformBlockNumbersInY} + (\log_2(\text{QuantizePrecision}[1][0] + 1) \times 1 +$$

$$+ \sum_{i=1}^{\log_2 nBlock} \log_2(\text{QuantizePrecision}[1][i]{+}1) \times 3 \times 4^{i-1}) \times \text{WaveletTransformBlockNumbersInCrCb}$$

$$\text{CompressionRatio} = \frac{\text{NewImageDataSize}}{\text{OriginalImageDataSize}}.$$

5 Entropy Encoding

The wavelet coefficients are now ready to be encoded. Each coefficient may be considered as a symbol. The domain of these symbols is [0, 255]. It may be different on the different color spaces and subbands. The goal of the entropy encoding stage is to minimize the average bit rate required to store all these symbols. This stage is lossless. Let X be a quantized wavelet coefficient, that takes its values among a finite set of QuantizePrecision + 1 symbols. The easiest way to store X is to use the fixed length encoding method. In this method, X can be coded by $\lceil(\log_2(\text{QuantizePrecision} + 1))$ bits. However, this simple encoding schema ignores one of the most important features of the quantized wavelets coefficients: lots of zeros and symbols with small values. To take advantage of this feature, traditional entropy encodings like the Run Length Encoding and Huffman encoding are applied. A special scan algorithm is implemented to optimize the entropy encoding.

5.1 Entropy-Encoding Schemas

Three entropy-encoding schemas can be applied. They are defined as ALLDATA, SUBBAND, and SPACE_SUBBAND in MinImage. The ALLDATA schema encodes the whole image data in one step. If the QuantizePrecision specified for each space and subband are the same, it is efficient to encode all these wavelet coefficients in one step. As the domains of these wavelet coefficients are the same, the SUBBAND schema divides the wavelet coefficients into subbands and encodes these subbands separately. The SPACE_SUBBAND schema encodes the every color space and every subband separately. The SPACE_SUBBAND encoding schema is suited in the condition that every QuantizePrecision is different for each color space and subband.

5.2 Scan The Wavelet Coefficients by Peano Curve

In each wavelet transform block, the coefficients are decompressed into several subbands. In order to entropy encode them efficiently, these coefficients are scanned by the Peano curve. An 8×8 wavelet transform block, scanned by the Peano curve is shown as Figure 16.

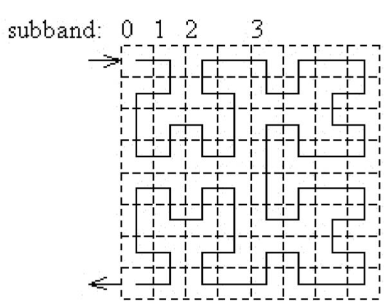


Figure 16. A Third Order Peano Curve

5.3 Entropy-Encoding

Figure 17 shows the entropy encoding schema in MinImage. A Run Length Encoding algorithm is used to compress the zeros in the current encoding unit if the entire elements in the unit are quantized by no more than six bits.

Zlib is a free compression library written in C and downloaded from the Internet. It compresses the source data by LZ77(Lempel-Ziv1977) and/or Huffman entropy encoding. Its deflation algorithms are similar with those used by PKZIP(an MSDOS based software by PKWARE, Inc.). A wrapped class is written to apply Zlib compression engine as the last stage of MinImage to compress all the wavelet coefficients to a final wavelet image file with the extension LET.

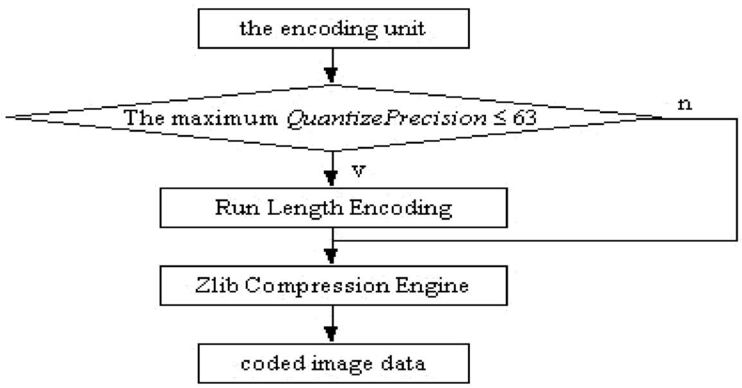


Figure 17 The Entropy Encoding Schema In MinImage

6 Constraints

MinImage is a complete wavelet image compression tool. There are some constraints of MinImage.

(1) The wavelets applied in MinImage are very basic ones. Hubbard noted in [6] that more complicated wavelets have already been proved to be more powerful to compress an image. Multiwavelets, wavelet packets, and multivariate wavelets are studied recently (see [2] – [5], [11], and the references therein). MinImage is coded in an object-oriented way so that it is relatively easier to implement the additional wavelets in the specific object.

(2) The entropy encoding is not the emphasized part of this paper. Therefore, traditional entropy encoding methods are used. An existing entropy-encoding library, Zlib, is imported to MinImage. There are two drawbacks. First, Zlib compressing engine may not be suited to compress the wavelet coefficients. Zlib is created as a general compressor. Consequently, it can not make all the advantages of the characteristics of the wavelets coefficients. Second, the entropy encoding is actually essential in the image compression applications. The efficiency of the entropy-encoding influences the final results in a significant way. Therefore, more sophisticated encoding strategies, such as the zero-tree encoding, studied by Hussain and Farvardin [7], or the vector quantization encoding, studied by Zhao and Yuan [12], should be implemented.

7 Conclusions

This paper represents wavelet applications in image compression, with the focus on the implementation details of MinImage as a wavelet image compressor. The three major components of MinImage are preprocessor, wavelet transform processor and the quantizator. Compared with the JPEG image compression standard, MinImage is a little better in some cases. In order to make MinImage a more powerful image compressor, entropy encoding methods can be improved greatly. As useful software, MinImage gives the user an easier way to do varieties of tests about the wavelet image compressions. Wavelets will be likely one of the best image compression standards in the future.

References

[1] Castleman, K.R., *Digital image processing.* Prentice Hall, New Jersey, (1996).

[2] Jiang, Q.T., Orthogonal multiwavelets with optimum time-frequency resolution, *IEEE Trans. Signal Process.* **46** (1998), 830–844.

[3] G. Plonka and V. Strela, From wavelets to multiwavelets, in *Math. Methods for Curves and Surfaces II*, (D. Dahlem, T. Lyche, and L. Schumaker eds.), Vanderbilt University Press (1998), 375–399.

[4] Hong, D. and Yuchun Anna Mu, On construction of minimum supported piecewise linear prewavelets over triangulations, In: "Wavelet Analysis and Multiresolution Methods", Lecture Notes for Pure and Applied Mathematics, pp.181–201, Marcel Dekker, Inc., New York, NY, 2000.

[5] Hong, D. and Aidi Wu, Multiwavelets with multiplicity four, *Computers and Mathematics with Applications*, to appear.

[6] Hubbard, B. B., *The world according to wavelets*, A K Peters Wellesley, Natick, MA, 1995.

[7] Hussain, Y. and N. Farvardin, *Variable-rate finite-state vector quantization and applications to speech and image coding*, (Technical Research Report No. 17), College Park, MD, University of Maryland, Systems Research Center, 1992.

[8] Mallat, S., *A wavelet tour of signal processing.* Academic press, San Diego, CA, 1998.

[9] Morton, P., and A. Petersen, *Image compression using the Haar wavelet transform*, (Tech. Rep. No. 3), College of the Redwoods, Department of Computer Science, Eureka, CA, 1997.

[10] Steinmetz, R. and K. Nahrstedt, *Multimedia computing, communications & applications*, Prentice Hall, New Jersey, 1997.

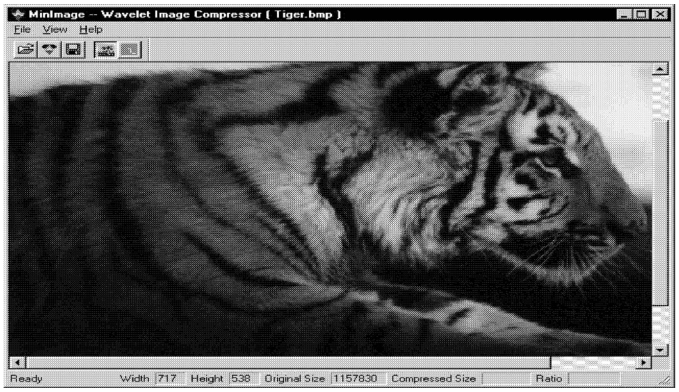
[11] Wang, J.Z., Interpolating cubic spline wavelet packet on arbitrary partitions, *J. Computational Analysis and Appl.* this issue, xx − xx.

[12] Zhao, Y. and B. Yuan, A new affine transformation: its theory and application to image coding, *IEEE Transactions on Circuits and Systems for Video Technology*, **3** (1998), xx-xx.

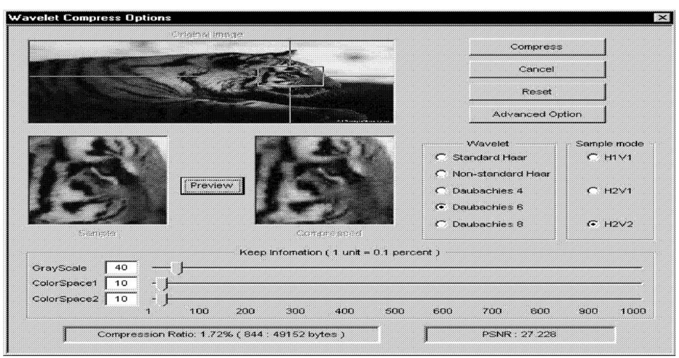
Appendix. Interface, Specification, and Analysis of MinImage

A Interface of MinImage

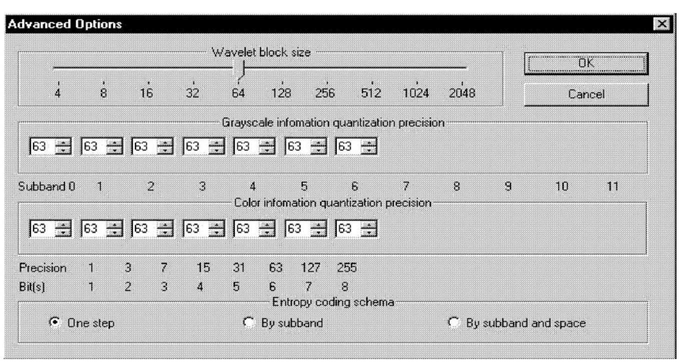
Here is the main window of MinImage.



The following is the compress interface. The user can easily preview the compressed result of a particular area in the original image without compressing the whole image.



Here is the interface of changing more advanced compression parameters in MinImage.



B Specification Of The Wavelet Compression Parameters

The parameters in the following table can be adjusted to compress the images in MinImage.

Parameter	Description	Domain
SampleMode	The mode of sampling the original image in the preprocessor.	0= H1V1 1 = H2V1 2 = H2V2
WaveletBlockBase	Log2WaveletTransformBlock Width (pixel width) This parameter decides the size of the wavelet transform blocks.	[1, 11]
WaveletType	The type of wavelets used in the discrete wavelet transform.	0 = Standard Haar 1 = Nonstandard Haar 2 = Standard Daub4 3 = Standard Daub6 4 = Standard Daub8
KeepCoefficientsPercent	The percentages of wavelet coefficients needed to be kept in each color space.	$[0.1\%, 100\%]$
QuantizePrecision	Each element in this vector defines the quantization precision of the wavelet coefficients in the specified color space and subband.	$[1, 255]$
EncodingStrategy	The strategy applied to entropy-encode the quantized wavelet coefficients.	0 = ALLDATA 1 = SUBBAND 2 = SPACE_SUBBAND

C Analysis

The analysis is focus on the fine tuning of wavelet transform parameters and the wavelet transform algorithm in MinImage.

The object image used in the tests is a star image, which is a 128×128 true type color image with 196,662 bytes. The default setting of the wavelet transform parameters are in the following Table.

THE DEFAULT SETTING OF WAVELET COMPRESSION PARAMETERS

Parameter	Default value
SampleMode	H2V2
WaveletBlockBase	7
WaveletType	DAUB4
KeepCoefficientsPercent	Grayscale: 2%, Colorspace1:0.5% Colorspace2:0.5%
QuantizePrecision	63 for all subbands in all color spaces
EncodingStrategy	ALLDATA

(1) The Optimization of the Wavelet Pyramidal Algorithm

In the pyramid wavelet decomposition algorithm, each repetition of the process divides the smooth data in half. The process, by theory, is proceeds until there are only two data points left. However, this process can be terminated at any point. The final output of the wavelet transform is the same number of data points as the input and could be used to reconstruct the original data, no matter how many step the wavelet transform is applied. One of the most important optimization of the MinImage is to terminate the wavelet transform in four steps. The setting of the wavelet transform parameter values is: $WaveletBlockBase = 8$. The rest wavelet transform parameters are set to the default values. When the wavelet transform is applied as the original pyramid algorithm, which involves $\log_2 n$ steps (full steps), the comparison ratio is 1.312When the wavelet transform is optimized, the transform terminates in four steps. The comparison ratio is 1.305%. It has much better image quality than the original result. The artificial effects are obvious in the original result. The conclusion is that less transform steps is better than full steps. The exact number of the step depends on different images and different wavelets. Experiments show that four or five steps are enough. MinImage implements four wavelet transform steps because four steps of wavelet transform schema works fine in most cases.

(2) Image Quality MinImage uses a popular image error measurement, PSNR (peak signal to noise ratio). It is based on the sum of the squared differences between corresponding pixels of two images. The exact formula is given in the following equation. $P_{r,c}$ is a pixel value in row r column c of image P and $Q_{r,c}$ is a pixel value in row r column c of image Q.

$$PSNR = 20 \times \log_{10} \frac{255}{\sqrt{\frac{\sum_{r=1}^{rows} \sum_{c=1}^{cols} (P_{r,c} - Q_{r,c})^2}{rows \times cols}}}.$$

PSNR measures the difference between two images. A bigger PSNR value usually means that two images compared have more similar pixels. Under normal circumstances PSNR is a good indicator of image quality, but both objective PSNR measurement and subjective observation measurement are needed to compare the image quality accurately.

(3) Choices of Wavelet

The default wavelet transform parameter values are set for the test. The results are represented in the following.

TESTING OF THE INFLUENCE OF WAVELET

WaveletType	PSNR	CompressionRatio	Subjective Quality
Standard Haar	30.688	1.076%	Non-acceptable
Non-standard Haar	30.728	1.072%	Not acceptable
Daub4	31.355	1.109% Acceptable	
Daub6	30.929	1.035%	Acceptable
Daub8	30.856	1.037%	Acceptable

There is little difference between standard and non-standard wavelet transforms, according to both the compression ratio and the image quality. The Daubachies wavelets are smoother than the Haar wavelet, therefore, Daubachies wavelets cause better compression ratio than the Haar wavelet.

(4) Reduce the Wavelet Coefficients

The default wavelet transform parameter values are set for the test.

underlineReduce the Grayscale Information

KeepCoefficientsPercent: For Colorspace1=100% and Colorspace2=100%, the results are represented in the following.

TESTING OF THE INFLUENCE OF REDUCE THE GRAYSCALE INFORMATION

KeepCoefficientsPercent Grayscale:	PSNR	CompressionRatio	Subjective Quality
5%	34.997	6.240%	Acceptable
4%	34.371	5.883%	Acceptable
3%	33.644	5.500 %	Acceptable
2%	32.760	5.066%	Acceptable
1%	31.424	4.523%	Non-acceptable
0.5%	28.353	4.250%	Non-acceptable

Reduce Both Colorspace1 and Colorspace2 Information

KeepCoefficientsPercent: For Grayscale=100%, the results are represented as the following.

TESTING OF THE INFLUENCE OF REDUCE THE COLOR INFORMATION

KeepCoefficientsPercent Corlorspace1, Corlorspace2	PSNR	Compression Ratio	Subjective Quality
5%	41.301	9.601%	Acceptable
4%	40.988	9.452%	Acceptable
3%	40.563	9.305%	Acceptable
2%	39.874	9.113%	Acceptable
1%	38.309	8.894%	Acceptable
0.5%	35.932	8.758%	Acceptable
0.3%	33.282	8.698%	Acceptable
0.2%	28.902	8.674%	Non-acceptable

The color information can be compressed more than the grayscale information. This is reasonable because human eyes are more sensitive of brightness than of colors.

(5) Wavelet Coefficients Quantization

The settings of the wavelet transform parameter values are: QuantizePrecision parameters are set to the same value to all color spaces and subbands. KeepCoefficientsPercent= Grayscale:4%, Colorspace1:1%, Colorspace2:1%. The rest wavelet transform parameters are set to the default values. The results are represented below.

TESTING OF THE INFLUENCE OF WAVELET COEFFICIENTS QUANTIZATION

QuantizePrecision	PSNR	CompressionRatio	Subjective Quality
255 (8bits)	33.663	2.731%	Acceptable
127 (7bits)	33.610	2.508%	Acceptable
63(6bits)	33.394	1.970%	Acceptable
31(5bits)	31.899	1.768%	Non-acceptable
15(4bits)	27.839	1.506%	Non-acceptable

Assigning 8 bits to quantize the wavelet coefficients will disable Run Length Encoding in MinImage. It is not necessary to use more than six bits to quantize the coefficients. The advantage of assigning the same quantization precision values to all the subbands in all color spaces are to enable entropy encoding all these coefficients in one step by ZLib compression Engine. By experiments, six bits are usually optimal to quantize the wavelet coefficients.

(6) Entropy Encoding Schema

The settings of the wavelet transform parameter values are: KeepCoefficientsPercent= Grayscale:4%, Colorspace1:1%, Colorspace2:1%. $WaveletBlockBase = 6$ The rest wavelet transform parameters are set to the default values.

Case 1: The *QuantizePrecision* parameters are set to the same default value to all color spaces and subbands. The results are represented in the table below.

It is better to compress all the wavelet coefficients in ALLDATA because the data in different subbands and spaces are quantized by the same number of bits.

Case 2: The QuantizePrecision parameters are set to the different value to the different subbands.

QuantizePrecision [subband0] = 255.

QuantizePrecision [subband1] = 127.

QuantizePrecision [subband2] = 63.

QuantizePrecision [subband$_i$] = 31 (any subband$_i$ > 2).

See the table below for the results.

In this case the SUBBAND entropy encoding strategy is the best option, because different QuantizePrecision values are set to different subbands, while the same QuantizePrecision values are set to different subband with same color space.

Case 3: The QuantizePrecision values are set to the different value to both the subbands and the color spaces. The grayscale space:

QuantizePrecision [subband0] = 255

QuantizePrecision [subband1] = 127

QuantizePrecision [subband2] = 63

QuantizePrecision [subband_i] = 31 (any $\text{subband}_i > 2$)

The color space:

QuantizePrecision [subband0] = 31

QuantizePrecision [subband1] = 63

QuantizePrecision [subband2] = 127

QuantizePrecision [subband_i] = 255 (any $\text{subband}_i > 2$)

The results are represented in the following together with the previous cases.

In this case the SPACE_SUBBAND entropy encoding strategy is the best option, because different QuantizePrecision values are set to both different subbands and different spaces.

TESTING OF THE INFLUENCE OF ENTROPY ENCODING SCHEMA

	Case 1	Case 2	Case 3
EncodingStrategy	CompressionRation	CompressionRation	CompressionRation
ALLDATA	2.001%	2.230%	2.328%
SUBBAND	2.002%	1.836%	2.285%
SPACE_SUBBAND	2.112%	1.959%	2.081%

(7) Conclusion of the Analysis

Different wavelet transform parameters are used under different conditions. The final compression ratio and image quality depends on not only the set of all these parameters, but also depends on the image itself. Therefore, there is no best solution to the setting of these wavelet transform parameters. The design of the MinImage is to make all these wavelet compression parameters easy to tune. By previewing the result of the compressed image, the user can always generate the image with particular quality and compression ratio. The flexibility of MinImage makes it a good tool to compress different image in different way.