1 of 1

TITLE: **1992 PROGRESS REPORT: COMPRESSION OF FINGERPRINT
DATA USING THE WAVELET VECTOR QUANTIZATION
IMAGE COMPRESSION ALGORITHM**

AUTHOR(S): Jonathan N. Bradley
Christopher M. Brislawn

SUBMITTED TO: For External Distribution

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

By acceptance of this article, the publisher recognizes that the U.S. Government retains a nonexclusive royalty-free license to publish or reproduce the published form of this contribution or to allow others to do so, for U.S. Government purposes.

The Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy.

Los Alamos

Los Alamos National Laboratory
Los Alamos New Mexico 87545

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

1992 PROGRESS REPORT:
COMPRESSION OF FINGERPRINT DATA USING THE
WAVELET VECTOR QUANTIZATION IMAGE
COMPRESSION ALGORITHM

by

Jonathan N. Bradley and Christopher M. Brislawn
Group C-3, Computer Research
Los Alamos National Laboratory

Submitted by Rosalind Newmyer, IT-6

April 11, 1992

1 INTRODUCTION.

This report describes the development of a Wavelet Vector Quantization (WVQ) image compression algorithm for fingerprint raster files. The pertinent work was performed at Los Alamos National Laboratory for the Federal Bureau of Investigation. This document describes a previously-sent package of C-language source code, referred to as LAFPC, that performs the WVQ fingerprint compression and decompression tasks. The particulars of the WVQ algorithm and the associated design procedure are detailed in [2]; the purpose of this document is to report the results of the design algorithm for the fingerprint application and to delineate the implementation issues that are incorporated in LAFPC. Special attention is paid to the computation of the wavelet transform, the fast search algorithm used for the VQ encoding, and the entropy coding procedure used in the transmission of the source symbols.

The WVQ algorithm is a special case of subband coding. This class of image coders involves a decomposition of a data signal into a number of subsignals (referred to here as *subbands*), each of which represents information in a particular frequency band. The subbands are quantized individually and subsequently recombined. By coding the subbands separately, higher signal-to-noise ratios are obtainable than by single-channel methods in general because the coding algorithm can be better matched to the signal statistics. In addition, since many perceptual criteria (both visual and auditory) are formulated in the frequency domain, these techniques allow for weighting the various frequency components so that the quantization noise can be matched to such criteria. In the WVQ algorithm, the subband decomposition is achieved by a wavelet transformation of the image. The wavelet transform is a recently developed mathematical technique that was strongly motivated by advancements in the development of subband coding filter banks in the 1980's. After the wavelet decomposition of the image, the subband compression is achieved by vector quantization (VQ). Although other techniques mentioned in the literature achieve image compression using VQ of wavelet transform coefficients, the WVQ algorithm is unique in two significant ways: (1) the wavelet transform realizes symmetric boundary conditions, and (2) the selection of the VQ codebooks proceeds with constraints on both the bit-rate and encoding complexity.

Section 2 discusses the subband decomposition used, the particular wavelet filters selected for the application, and the incorporation of symmetric boundary conditions into the WVQ scheme. Section 3 reviews VQ; of particular interest is the discussion of the rate-distortion properties of this type of quantizer. Section 4 presents the intraband SBVQ design algorithm. Section 5 addresses the method whereby the source symbols generated by the vector encoders are entropy coded for transmission/storage. In Section 6 an important aspect of LAFPC is covered: the k-dimensional search technique for fast VQ encoding. Section 7 documents experimental results, including the vector dimensions and codebook size assignments generated by the design algorithm, distortion measurements of actual compressed images, and visual comparisons of uncompressed and compressed fingerprint raster files. Section 8 discusses the C-language source code, LAFPC, and includes instructions on program usage.

Figure 2.1: Two-Channel Subband Coder.

2 THE DISCRETE WAVELET TRANSFORM.

2.1 The Two-Channel Subband Coder.

A two-channel subband coder is a digital filter bank of the type shown in Figure 2.1. For image processing applications, the input data, x , will consist of a row or column of pixels from a digitized image; for now, we will simply regard x as a sequence of N real numbers: $x(n)$; $n = 0, \dots, N-1$. The complex conjugations on the frequency responses of the analysis filters indicate that analysis is done by correlating x with the filters h_0, h_1 ; i.e., the k^{th} output of H_i^* is

$$\sum_n x(n) h_i(n-k); \quad i = 0, 1.$$

(We will defer giving explicit limits for the summation until Section 2.4.2, when we'll discuss boundary conditions for finite-length signals.)

The 2:1 downsampling operator throws out all the odd terms in a sequence,

$$(2 \downarrow y)(k) = y(2k),$$

so the output of the analysis bank consists of the two sequences

$$a_i(k) = \sum_n x(n) h_i(n-2k); \quad i = 0, 1,$$

each half as long as x . We will refer to the transformation $x \mapsto \{a_0, a_1\}$ as a *discrete wavelet transform* (DWT).

The synthesis bank in Figure 2.1 starts with an upsampling operator, which inserts zeros between the entries in a sequence:

$$(2 \uparrow a)(k) = \begin{cases} a(k/2), & k \text{ even} \\ 0, & k \text{ odd.} \end{cases}$$

This is followed by convolution with one of the synthesis filters $\tilde{h}_0, \tilde{h}_1$, and the resulting sequences are then added. Thus, the synthesis calculation is

$$\tilde{x}(n) = \sum_k a_0(k) \tilde{h}_0(n-2k) + \sum_k a_1(k) \tilde{h}_1(n-2k).$$

Figure 2.2: Subband Frequency Supports.

Figure 2.3: Two-Level Wavelet Subband Analysis.

We shall only consider systems of this type that result in perfect reconstruction of x ; i.e., $\tilde{x} = x$. (This is, of course, assuming no quantization distortion in the DWT subbands a_0 and a_1 .) The transformation $\{a_0, a_1\} \leftarrow x$ is called the *inverse* DWT.

Throughout, we shall let h_0 be a low-pass finite impulse response (FIR) filter and h_1 a high-pass FIR filter. The frequency supports of the subbands a_0 and a_1 are approximately as shown in Figure 2.2. (“Approximately” means that the frequency separation is not absolute because h_0 and h_1 are not ideal bandpass filters. In spite of this, it’s usually simplest to ignore frequency roll-off and just think of the ideal case.) For instance, the segment from 0 to $\pi/2$ labeled “ a_0 ” represents the $N/2$ numbers $a_0(n)$; $n = 0, \dots, N/2 - 1$, and indicates that these numbers are coefficients of a signal component (the output of the low-pass synthesis channel) whose discrete Fourier transform (DFT) is concentrated on the interval $[0, \pi/2]$.

We will form more complicated subband decompositions by cascading the output channels a_0 and a_1 back through the DWT. The outputs of these extra levels of transformation will be indicated with additional subscripts. For instance, the results of running a_0 through

Figure 2.4: Second-Level Subband Frequency Supports.

Figure 2.5: Third-Level Subband Frequency Supports.

the analysis bank will be the subbands a_{00} and a_{01} , each only one quarter as long as x ; running a_1 through the analysis bank produces bands a_{10} and a_{11} . (See Figure 2.3.)

The frequency support picture of a cascaded tree-coder like this is complicated by a phenomenon known as *spectral inversion* [4]. It can be shown that the roles of the high-pass and low-pass filters in a second-level decomposition are *interchanged* when they follow a first-level high-pass filter. Thus, the h_0 channel in the second level of Figure 2.3 has the effect of a *high*-pass filter when applied to a_1 , but the effect of a *low*-pass filter when applied to a_0 . Approximate frequency supports for the subbands in Figure 2.3 are indicated in Figure 2.4.

For a third-level decomposition, we will have spectral inversion in the two bands that have had only one h_1 operation so far; i.e., bands a_{01} and a_{10} . The second h_1 filtering in band a_{11} had the effect of a low-pass filter, so there will be no third-level spectral inversion when splitting a_{11} . Figure 2.5 shows the approximate frequency supports for a full three-level tree coder. In general, the coder always undergoes spectral inversion following a high-pass operation, regardless of whether the high-pass filtering was done by h_1 or h_0 . Thus, e.g., we will still have spectral inversion in a fourth-level splitting of a_{100} because the application of h_0 in the third level had the effect of a high-pass filter.

2.2 Wavelet Decompositions in Two Dimensions.

Next, we extend these DWT signal processing techniques for two-dimensional discrete signals, $x(m, n)$, such as digitized images. The extension is based on (tensor) products of the one-dimensional DWT described in the previous section. Computationally, this means applying the analysis bank in Figure 2.1 to the row vectors in x , then to the column vectors of the resulting output. A single-level decomposition of a two-dimensional signal is shown in Figure 2.6; note how we use a *pair* of binary indices to describe the filters applied to

Figure 2.6: Single-Level, Two-Dimensional Wavelet Subband Analysis.

Figure 2.7: Single-level, Two-Dimensional Wavelet Subband Synthesis.

the rows and columns of x . Thus, e.g., a_{10} results from applying h_1 to the row vectors of x and h_0 to the column vectors of the results of the row transform. Single-level synthesis is shown in Figure 2.7, and approximate two-dimensional frequency supports are shown in Figure 2.8.

Although we have adopted the convention of always applying row operations first, row and column operations can in fact be done in either order, yielding the same result modulo roundoff error. This implies that spectral inversion effects can be computed independently for row and column operations in two dimensions. Figure 2.9 shows approximate spectral supports for three second-level splittings involving the a_{00} , a_{01} , and a_{10} bands. The letter “ a ” has been omitted for simplicity, as have the digits “00” for leading low-pass filterings; e.g., “ $,10$ ” is for shorthand for “ $a_{00,10}$.” Note how spectral inversion occurred only in the row operations (indicated by the first bit in each pair) on band 10 and only in the column operations (the second bit) on band 01.

2.3 Subband Design for Fingerprints.

A key problem in designing a DWT subband coder for fingerprint compression is the choice of a particular subband decomposition. Rather than pursuing a more computationally expensive adaptive approach, as is used in the Best-Basis algorithm, we believe that fingerprint data is statistically homogeneous enough to permit designing a single subband decomposition optimized for the fingerprint ensemble. The present version of the WVQ code, LAFPC-2, uses a straight octave-scaled, four-level DWT decomposition: Figures 7.3 and 7.4, respectively, show a 768×768 fingerprint rasterfile and the corresponding transform. This choice was based more on prevailing convention and ease of programming than on rigorous analysis of fingerprint data. Examination of the performance results from LAFPC-2 indicates that there is room for improvement, so we now discuss a couple data analysis techniques that offer some insight into more intelligent design approaches.

Since the subband design is to be based on ensemble statistics, we consider two different sources of information based on averages over a number of different fingerprints: 1) a histogram power spectral density (PSD) estimate, and 2) a mean information-minimizing decomposition (IMD). These two different signal analyses are connected by our understanding of the spectral inversion phenomenon, which enables us to relate DWT spectra to Fourier spectra.

2.3.1 Fourier PSD Estimation.

This section describes the computation of a two-dimensional PSD estimate for fingerprint data. The estimate is based on 50 records consisting of all 10 fingers from the following cards.

107879.*m
155309.*d
155498.*m
628663.*m
757631.*d

Figure 2.8: Two-Dimensional Subband Frequency Supports.

Figure 2.9: Two-Dimensional Spectral Inversion.

These records were chosen for their high-quality images. Each print was cropped to 512×512 pixels to eliminate blank card, handwriting, bounding lines, text, etc., so as to estimate only the information we're interested in preserving.

After cropping, the ℓ^2 norms of the records were equalized by rescaling to 512. The mean of each record was then calculated and subtracted off to provide mean-zero data. Since our DWT uses reflected—rather than periodized—boundary conditions, we computed the 1024-point DFT of the symmetric extension of each record using a 512-point discrete cosine transform (DCT). The 512×512 DCT's were squared and then averaged over the 50-record collection. For the purpose of generating reasonably legible graphics, the 512×512 histogram PSD was reduced to 128×128 by averaging 4×4 blocks of numbers. This smoothed histogram estimate is displayed in Figures 2.10 and 2.11 with logarithmically scaled amplitudes. Note that only the low-frequency quadrant, $[0, \pi/2] \times [0, \pi/2]$, is shown.

The prominent “ridge” visible in the surface plot in Figure 2.10 is evidently the spectral peak corresponding to fingerprint ridges. Note the asymmetric distribution of power along this peak. More detail can be discerned in the contour plot in Figure 2.11. The fingerprint ridge-width peak is clearly visible in the portion of the spectrum lying roughly between $\omega = \pi/4$ and $\omega = \pi/8$; this corresponds to signal components with periods of 8 to 16 pixels, which is consistent with the fingerprint ridge-widths observed in the images. The asymmetric power distribution shows more power in this interval in the vertically transformed part of the spectrum (ω_{col}) than in the horizontally transformed part (ω_{row}). This may be due to the fact that fingerprints are aligned vertically on the card, which may mean that a typical column of pixels traverses a greater number of ridges than the typical row. Whatever the cause, to date we have neglected this asymmetry in our coder designs and assumed that the fingerprint ensemble is rotationally invariant for the purposes of designing the subband decomposition and the individual subband quantizers.

Two more features emerge from examination of the power spectrum. First, the PSD appears to be a featureless desert for frequencies much higher than about $\pi/4$. All we can see in this regime is a gradual, radially symmetric decay of power as frequencies increase to π . (For this reason, we have only shown the first quadrant of the PSD.) This is also consistent with the results of the optimal VQ bit allocation program, which discards bands a_{10} , a_{01} , and a_{11} altogether based on minimum mean-square distortion criteria. From this observation, we conclude that signal power (or variance) measurements alone cannot detect features occurring at frequencies above $\omega = \pi/4$; i.e., on length scales of less than about 8 pixels. This includes transient features like islets and sweat pores, which are typically on the order of 2-4 pixels in diameter. Indeed, scanner noise in poorly scanned images is comparable in power to the legitimate information in high-passed DWT subbands; examine band 01 in Figure 7.4, which reveals significant row-to-row scanner bias in the image. This suggests that band 01 should not be preserved without additional levels of frequency partitioning.

Second, the power trough between the DC peak and the ridge-width peak in the PSD contains a number of points at frequencies below $\pi/16$. Since the approximate frequency support of the fourth-level DWT low-band is $[0, \pi/16]$, this band contains information about features at frequencies ranging from DC clear up into the power trough. Therefore, it seems advisable to separate these features by splitting off a fifth-level DWT low-band. The size of a fifth-level subband from a 768×768 original is only 24×24 , though, so it's not clear that

Figure 2.10: Two-Dimensional PSD Estimate of Fingerprint Data.

Figure 2.11: Contour Plot of Two-Dimensional PSD Estimate of Fingerprint Data.

there is much additional compression to be had from further subdivisions. There seems to be no need to incur additional computational complexity in the low-band in light of the fact that the shortcomings of the current algorithm lie in the extremely high frequency regime.

2.3.2 The Mean Information-Minimizing Decomposition.

Our second tool for analyzing different DWT subband decompositions is an estimate of the information-minimizing decomposition (IMD) for the average fingerprint. The software for computing this decomposition is less refined than the PSD program described above, and the mean IMD presented here is based on only 8 records.

In an attempt to replicate FBI findings, the measure of information we used in a given band was the number of coefficients exceeding 12 in absolute value. To determine whether a band should be subdivided, the band was split and the sum of the information content in the four resulting subbands computed. The splitting was retained as part of the decomposition if it resulted in a reduction in information as computed by the above measure. Each band was tested in this fashion until no further decreases in information were observed, down to at most 6 levels of subdivision. Our interpretation of this information-reducing phenomenon is that significant information initially given in the form of large coefficients is being transformed into information encoded in the structure of a more complex subband decomposition.

The IMD was computed as above for 512×512 cores of the following eight images.

103750.4d	757631.6d
155309.6m	573288.2m
114439.8m	107879.1d
772983.2m	155498.1d

We found that we obtained much finer (more complex) decompositions for 512×512 cores than we did for full-sized 768×768 images; we conjecture that this is due to energy in the white background overshadowing comparatively smaller changes in high-frequency detail information. We also observed a large variation—by a factor of about 4—in the number of subbands in the IMD's for these eight images.

Figure 2.12 shows all the subband splittings, ordered by approximate spectral supports, that occurred in at least four of the eight records, a subband decomposition we call the mean IMD. In addition, because of our premise that encoder design should not be orientation-specific, we have also included subdivisions of bands 10 and 10.10 although these two particular splittings occurred in only three of the eight IMD's. While this is admittedly a crude measure of the distribution of information in the wavelet spectrum, it does provide some additional guidance in the choice of DWT decompositions.

2.4 Computation of the DWT.

We now consider the problem of computing the DWT near the ends of a finite-length signal. One could periodize the signal and compute the DWT using circular convolution, but this method has the disadvantage of introducing a large jump discontinuity into the signal where the (usually uncorrelated) ends are matched up. For this reason, we have developed an approach based on transforming a periodized, even extension of the signal.

Figure 2.12: Mean Information-Minimizing Decomposition for Fingerprints.

The next two sections will concentrate on the transform of greatest interest to date, the reflected DWT based on a family of biorthogonal wavelet filters constructed in [3].

2.4.1 Biorthogonal Wavelet Family b4-4.

The filters we have been using to date come from the polynomial

$$P_4(x) = \sum_{n=0}^3 \binom{3+n-1}{n} x^n.$$

We have the factorization

$$P_4(\sin^2 \omega/2) = p(\cos \omega) \tilde{p}(\cos \omega),$$

with p irreducible quadratic and $\tilde{p}$ monic, normalized so that $p(\cos 0) = 1$. The dual low-pass filters are given in the frequency domain by

$$\begin{aligned} H_0(e^{i\omega}) &= \sqrt{2} \cos^4(\omega/2) p(\cos \omega) \quad \text{and} \\ \tilde{H}_0(e^{i\omega}) &= \sqrt{2} \cos^4(\omega/2) \tilde{p}(\cos \omega). \end{aligned}$$

Both of these filters have impulse responses, h_0 and $\tilde{h}_0$, that are symmetric about 0 with 9 and 7 coefficients, respectively. The corresponding high-pass filters, which are symmetric about 1, are

$$\begin{aligned} h_1(n) &= (-1)^n \tilde{h}_0(1-n) \quad \text{and} \\ \tilde{h}_1(n) &= (-1)^n h_0(1-n). \end{aligned}$$

Analytic and approximate values for h_0 and $\tilde{h}_0$ are given in Table 2.1. The scaling functions corresponding to these low-pass filters are shown in Figures 2.13 and 2.14; the mother wavelets for the high-pass filters are in Figures 2.15 and 2.16.

2.4.2 Reflected Boundary Conditions for the DWT.

We now describe a DWT based on forming an even extension of the signal; i.e., reflected boundary conditions. The transform conserves storage costs in the sense of transforming an N -point sequence, x , into two subbands each of length $N/2$. To accomplish this, we need to use a symmetric extension, y , of x for which the resulting DWT subbands will also be symmetric.

Consider a finite length signal $x(n); n = 0, \dots, N-1$, as suggested in Figure 2.17 (with $N = 4$.) There are various symmetric extensions of x that one might consider; three such extensions are shown in Figure 2.18(a-c). The difference between these extensions is the matter of whether the endpoints of the signal are duplicated in the extension. We use a pair of digits, (i, j) , to indicate whether the left/right endpoint of x is duplicated in the extension. If the respective endpoint is duplicated, then i or j is set equal to 2; a value of 1 indicates an unduplicated endpoint. Thus, e.g., the extension $y = x^{(1,2)}$ in Figure 2.18(b) has a duplicated right endpoint and an unduplicated left endpoint. All three extensions are to be regarded as one full period of an infinite, periodic signal. (A second period of each extension is shown in Figure 2.18 with hollow dots.)

Table 2.1: Lowpass Wavelet Filters.

Tap	Exact Value	Approx. Value
$h_0(0)$	$-5\sqrt{2}x_1(48 x_2 ^2 - 16\Re x_2 + 3)/32$	0.85269867900940
$h_0(\pm 1)$	$-5\sqrt{2}x_1(8 x_2 ^2 - \Re x_2)/8$	0.37740285561265
$h_0(\pm 2)$	$-5\sqrt{2}x_1(4 x_2 ^2 + 4\Re x_2 - 1)/16$	-0.11062440441842
$h_0(\pm 3)$	$-5\sqrt{2}x_1(\Re x_2)/8$	-0.023849465019380
$h_0(\pm 4)$	$-5\sqrt{2}x_1/64$	0.037828455506995
$\tilde{h}_0(0)$	$\sqrt{2}(6x_1 - 1)/16x_1$	0.78848561640566
$\tilde{h}_0(\pm 1)$	$\sqrt{2}(16x_1 - 1)/64x_1$	0.41809227322221
$\tilde{h}_0(\pm 2)$	$\sqrt{2}(2x_1 + 1)/32x_1$	-0.040689417609558
$\tilde{h}_0(\pm 3)$	$\sqrt{2}/64x_1$	-0.064538882628938

where:

$$\begin{aligned}
 x_1 &= A + B - 1/6 \\
 x_2 &= -\frac{A+B}{2} - 1/6 + i\sqrt{3}(A-B)/2 \\
 A &= \left(\frac{-14\sqrt{15} + 63}{1080\sqrt{15}} \right)^{1/3} \\
 B &= \left(\frac{-14\sqrt{15} - 63}{1080\sqrt{15}} \right)^{1/3}
 \end{aligned}$$

Figure 2.13: Scaling Function for Lowpass Analysis Filter h_0 .

Figure 2.14: Scaling Function for Lowpass Synthesis Filter h_0 .

Figure 2.15: Mother Wavelet for Highpass Analysis Filter h_1 .

Figure 2.16: Mother Wavelet for Highpass Synthesis Filter $\tilde{h}_1$.

Figure 2.17: A Finite-Length Discrete Signal.

Figure 2.18: Three Even, Periodic Extensions of x .

The correct extension of x to use in conjunction with odd-length symmetric filters like those described in Sec. 2.4.1 is the (1,1)-symmetric extension, $y = x^{(1,1)}$, shown in Figure 2.18(a):

$$x^{(1,1)}(n) = \begin{cases} x(n); & n = 0, \dots, N-1 \\ x(2N-2-n); & n = N, \dots, 2N-3. \end{cases}$$

Note that this extension is $(2N-2)$ -periodic. The calculation of filter outputs can now be done with $(2N-2)$ -periodic circular correlation:

$$\sum_{n=0}^{2N-3} y(n)h_i(n-k); \quad i = 0, 1,$$

where we regard the filters h_i as also being $(2N-2)$ -periodic, e.g., $h_i(2N-3) = h_i(-1)$. Similarly, we regard the synthesis filters $\tilde{h}_i$ as being $(2N-2)$ -periodic and compute the synthesis operations by circular convolution.

The 2:1 downsampling in each analysis channel leaves two output sequences, b_0 and b_1 , of period $N-1$:

$$b_i(k) = \sum_{n=0}^{2N-3} y(n)h_i(n-2k); \quad k = 0, \dots, N-2; \quad i = 0, 1.$$

This transformation has not increased the amount of information we need to store, however, because the sequences b_0 and b_1 each contain only $N/2$ distinct numbers. The $(N-1)$ -periodic sequences b_0 and b_1 have the following symmetries:

- a) if h_0 is symmetric about 0, then b_0 is (1,2)-symmetric;
- b) if h_1 is symmetric about 1, then b_1 is (2,1)-symmetric.

In both cases, the coefficient sequence b_i can be reconstructed from its first $N/2$ terms. This furnishes us with a reflected DWT that maps an N -point sequence, x , to two coefficient sequences of length $N/2$ each. Compare the following computations to Figure 2.1.

Analysis. Form the $(2N-2)$ -periodic extension $y = x^{(1,1)}$. For $i = 0, 1$ compute

$$a_i(k) = \sum_{n=0}^{2N-3} y(n)h_i(n-2k); \quad k = 0, \dots, N/2-1.$$

Note that this usually involves far fewer than $(2N-2)$ multiplications and additions per coefficient because h_i contains only a small number of nonzero terms.

Synthesis. Form the $(N-1)$ -periodic extensions

$$b_0 = a_0^{(1,2)} \quad \text{and} \quad b_1 = a_1^{(2,1)}.$$

The synthesis calculation for $x(n)$, $n = 0, \dots, N-1$, is

$$x(n) = \sum_{k=0}^{N-2} b_0(k)\tilde{h}_0(n-2k) + \sum_{k=0}^{N-2} b_1(k)\tilde{h}_1(n-2k).$$

Again, this computation only involves a small number of arithmetic operations, determined by the number of nonzero terms in the filters $\tilde{h}_0$ and $\tilde{h}_1$.

3 VECTOR QUANTIZATION.

A vector quantizer [8, 10, 12] consists of an encoder-decoder pair defined by an N element codebook of k -dimensional vectors. The bit-rate of the quantizer (the number of bits required for transmission of each input sample) is

$$r = \frac{\log_2 N}{k}. \quad (1)$$

The encoder subdivides the discrete-time input signal into k -dimensional *data vectors* $\mathbf{x}_n$ and then (ideally) for each $\mathbf{x}_n$ determines the codevector $\mathbf{y}_i$ that is most representative. This is equivalent to determining $\mathbf{y}_i$ such that

$$d(\mathbf{x}_n, \mathbf{y}_i) \leq d(\mathbf{x}_n, \mathbf{y}_j) \quad \forall j \neq i,$$

where $d(\mathbf{x}, \mathbf{y})$ denotes some measure of distortion between two vectors. When mean-square error (MSE) between the input and quantized signals is of interest, the appropriate distance measure is squared Euclidean distance,

$$d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^k (x_i - y_i)^2. \quad (2)$$

A straightforward method for implementing the encoder is to compute the distance between the input vector and each element of the codebook and then select that codevector corresponding to the minimum of these distances. For the distance measure (2) this entails N distance computations, resulting in Nk multiplications and approximately $2Nk$ additions for each input vector. Denoting the complexity of performing two adds and one multiply as α and assuming that the cost of performing the comparisons is negligible, the complexity Q of encoding one data sample is then

$$Q \approx 2^{rk} \alpha.$$

The decoder, which consists simply of a look-up table, has knowledge of the same codebook used by the encoder and constructs the quantized signal from the received codevector indices. The complexity of this process is trivial compared to that of the encoder. It is the encoding complexity that constrains the selection of k and N .

The codebook is designed to minimize the expected value of the quantization distortion. The design algorithm operates iteratively on a set of training data and is guaranteed to converge to a locally optimal solution [10]. This method is often referred to as the ‘‘Generalized Lloyd Algorithm’’ since it is a multidimensional generalization of Lloyd’s method II for PCM design [11].

An accurate description of how the quantization distortion relates to the bit-rate is necessary in this work since the task of designing a family of vector quantizers for the subbands requires knowledge of how resource allocation among the various subbands affects the distortion in the quantized image. The VQ distortion-rate characteristic is assumed to be given by

$$D(k, R) = \beta(k) e^{-\gamma(k)r}, \quad (3)$$

where $\beta(k)$ and $\gamma(k)$ are determined empirically.

4 THE WVQ DESIGN ALGORITHM.

The problem of designing a set of vector quantizers for coding a wavelet decomposition involves selecting the vector dimension, k , and codebook size, N , for each subband. Once these parameters are specified, each codebook can be obtained by applying Lloyd's algorithm to an appropriate set of training data. The method for selecting these parameters involves an extension of the reasoning used in [14] in which the bit-rate of the coder for each vector component was obtained via an optimization problem that specified that the bit-rates be selected to minimize the overall distortion of the coding scheme while maintaining some specified overall bit-rate. However, if only a rate constraint is imposed for a WVQ system, the optimization algorithm will select the vector dimension of each vector quantizer arbitrarily large. For this reason, a constraint on the overall encoding complexity is included in the problem formulation. The problem statement for WVQ system design is as follows. Minimize

$$D = \sum_{i=1}^M \frac{1}{m_i} \beta_i(k_i) e^{-\gamma_i(k_i)r_i} \quad (4)$$

over the k_i and r_i , subject to

$$\sum_{i=1}^M \frac{r_i}{m_i} \leq R, \quad (5)$$

$$\sum_{i=1}^M \frac{1}{m_i} 2^{r_i k_i} \leq Q, \quad (6)$$

$$r_i \geq 0, \quad \text{and} \quad (7)$$

$$k_i \in K_i. \quad (8)$$

m_i is the downsample factor for the i^{th} DWT subband; e.g., $m_i = 4$ for first-level subbands because they're one-fourth the size of the original image. Equation (4) is the objective function; it describes how distortion in the subband coder is quantified. In the case of an orthogonal transform, this quantity describes MSE. Equations (5) and (6) are, respectively, the rate and encoding complexity constraints. Equation (7) is included to force the channel bit-rates to be nonnegative. The constraint (8) restricts $\mathbf{k}$ to be an element of a finite set. This is done because the β_i and γ_i can only be obtained for a finite set of values. The r_i are necessarily discrete since a vector quantizer must have an integral number of codevectors. Nonetheless, for simplicity, it is assumed that the r_i are continuous; after a solution is found, the N_i are calculated by

$$N_i = \lceil 2^{r_i k_i} \rceil,$$

where $\lceil \cdot \rceil$ denotes the function that maps a real number to the closest integer. It is impractical, however, to treat the k_i as continuous variables.

Due to the presence of both continuous and discrete variables, the optimization problem is separated into two subproblems: one dealing with the selection of the optimal values for the discrete variables and the other dealing with the continuous variables. The rate optimization subproblem assumes that the k_i are fixed; the solution involves nonlinear programming techniques and is based on the program NPSOL [7]. This subproblem is a minimization of a convex function over a convex set; such problems are known to possess

a unique local minimum. The subproblem concerned with the optimization of the k_i is a heuristic pattern search. The overall system design problem is discussed in detail in [2].

5 ENTROPY CODING OF SOURCE SYMBOLS.

In the development of the design algorithm discussed in Section 4, the VQ bit-rate was described by (1), an equation based on the assumption that each codeword for a set of N source symbols is of length $\log_2 N$ bits. This actually represents an upper bound on the quantizer bit-rate, with equality being achieved in the case where the source symbols occur with equal probability. The well-known technique of Huffman coding [13] exploits an unequal distribution in source symbol probabilities by constructing a variable-length code in which symbols of lesser probability are assigned longer codewords. This technique achieves a bit-rate that is bounded below by the source entropy.

Huffman coding, however, does not exploit any correlations that exist among successive source symbols. The LAFPC-3 software includes a modification to the basic WVQ algorithm in which the transform coefficients are modified before VQ to create strong intersymbol dependencies in areas of low image detail; this can in turn be used to improve entropy coding gain. This is accomplished by selecting a threshold value for each subband and zeroing all wavelet coefficients with absolute value less than this quantity. Since this thresholding introduces additional distortion over the quantization noise, the threshold values should be selected based on a perceptual criterion such that the subjective distortion resulting from this procedure is minimal. The thresholding results in long strings of zero coefficients in the transform output in areas with very little detail, which in turn results in long strings of a particular source symbol. In this work, a set of threshold values for the four-level octave-scaled frequency decomposition was chosen empirically with the goal of zeroing as large a portion of each subband as possible without introducing unacceptable distortion. The distortion from excessive threshold values takes the form of "haloing" around handwriting and the fingerprint boundary. The threshold values are tabulated in Table 5.1. Lloyd's algorithm was modified slightly to account for the threshold effects: each codebook was forced to have a zero vector to code the thresholded regions and this vector was assigned index 0. Hence, long runs of zeros are present in the codevector indices of each subband due to areas in the card where the fingerprint is absent.

A zero run-length Huffman coder was used to entropy code the modified wavelet coefficients. This method involves the use of a traditional Huffman coder to code occurrences of both zero runs as well as individual source symbols. Terminal nodes were included in the Huffman tree to represent zero run-lengths of size 2, 32, 64, 128 and 256. Consequently, for N source symbols, there are $N + 34$ terminal nodes in the corresponding Huffman coding tree. Zero run-lengths of a size that does not appear in the table are described as a series of runs of sizes that do appear, e.g., a run of 75 zeros is described as a run of 64 followed by a run of 11.

Table 5.1: Threshold Values for the Four-Level Octave-Scaled Decomposition.

Subband Number	Subband Name	Threshold Value
0	...,00	0.
1,2	...,01 ...,10	30.
3	...,11	30.
4,5	.,01 .,10	25.
6	.,11	25.
7,8	.01 .10	15.
9	.11	15.
10,11	01 10	5.
12	11	2.

6 THE K-D TREE FAST SEARCH ALGORITHM.

The VQ encoding procedure was mentioned in Section 3. For each input vector, the task entails finding the codebook element that is nearest in Euclidean distance. Efficient procedures for this problem (commonly referred to as the “Nearest Neighbor” problem) have been widely researched because it is widely encountered in pattern recognition applications.

An efficient and well-known algorithm for accomplishing the search was introduced in [5]. A binary tree is utilized in which each node is characterized by a subset of the codebook and a key value that defines a division of k -dimensional space by a partition along one of the k axes. The key values effectively define a grouping of k -dimensional space into cubes. The root node represents the entire codebook, and each successive node represents a further codebook division, corresponding to the subregion of space defined by the key of its predecessor. An input data vector is encoded by traversing the tree in a manner defined by the key at each node encountered. Upon reaching a terminal node (referred to as a *bucket*), the algorithm performs a full search of the appropriate subset of the codebook, makes some tests as to whether or not the closest vector in the bucket is the closest for the entire codebook, and then may travel back up the tree and down to other buckets. Although the algorithm performs a full-search in the buckets encountered, the overall performance is good because the subset of the codebook that is searched is relatively small. The k-d tree search entails a complexity that is linear with rk , as compared to the full-search method whose complexity is exponential with rk .

7 EXPERIMENTAL RESULTS.

This section documents the results of the WVQ design algorithm for the fingerprint application and demonstrates the resulting system performance by tabulating distortion-rate measurements and presenting visual comparisons of original and compressed fingerprints.

The initial step in the design phase is the estimation of the $\beta_i(k)$ and $\gamma_i(k)$ that appear in the optimization objective function (4). For given i and k , this is done in two steps: (i) designing a set of codebooks of various sizes and then (ii) encoding a set of training data

Figure 7.1: VQ Distortion-Rate Measurements for Subband „10.

(different than the one used for the codebook design) and recording the resulting mean-square error in each case. These outcomes provide information from which the $\beta_i(k)$ and $\gamma_i(k)$ can be estimated by linear regression techniques. Figure 7.1 illustrates the process for the case of $k = 4$ for subband „10. Codebooks consisting of 2, 4, 8, 16, 32, 64, 128, 256, 512 and 1024 4-dimensional codevectors were designed for subband „10 using a set of training images. A separate set of data was then encoded with all of these codebooks; distortion measurements were made in each case and are depicted on a log-linear scale in Figure 7.1. The data values have a near-linear characteristic, which motivated the selection of the exponential function (3) to model VQ distortion-rate behavior. A linear regression of the data points is also shown; the slope and intercept of this line determine the values of $\gamma_i(4)$ and $\beta_i(4)$. The values of the $\beta_i(k)$ and $\gamma_i(k)$ computed for the four-level octave-scaled decomposition are listed in Tables 7.1 and 7.2, respectively. We assumed that fingerprint data are isotropic, and hence, the VQ rate-distortion parameters for appropriate subbands were constrained to be equal (e.g., the parameters for subbands „01 and „10 are the same). The vector dimension and codebook-size selections generated by NPSOL for various encoding complexities are tabulated for $R = 0.4\text{bpp}$ (20:1 compression) in Table 7.3 and for $R = 0.5$ (16:1 compression) in Table 7.4. As in the two-dimensional PSD estimation discussed in Section 2, 512×512 sections of fingerprint raster files were used so that the codebooks would be tuned to fingerprint details. However, later in the design procedure, where probability tables of the source symbols are estimated, full-sized 768×768 images are used.

Table 7.1: $\beta_i(k)$ for Four-Level Octave-Scaled Decomposition.

i	Subband	$k = 1$	$k = 2$	$k = 4$	$k = 8$
0	...00	850723.9	333106.3		
1,2	...01 ...10	18497.35	21085.41		
3	...11	24014.11	24812.11		
4,5	..01 ..10	14054.97	19388.31	18095.02	
6	..11	6438.083	5921.347	5485.404	
7,8	.01 .10	795.8949	1132.627	953.0213	
9	.11	170.6023	251.9025	219.8907	
10,11	01 10	33.98008	43.8642	36.34755	33.79507
12	11	4.284856	3.693484	3.053434	2.812638

Table 7.2: $\gamma_i(k)$ for Four-Level Octave-Scaled Decomposition.

i	Subband	$k = 1$	$k = 2$	$k = 4$	$k = 8$
0	...00	1.37552	1.470854		
1,2	...01 ...10	0.770843	1.021286		
3	...11	0.715238	0.974309		
4,5	..01 ..10	0.673317	0.974212	1.161308	
6	..11	0.783847	0.975050	1.152711	
7,8	.01 .10	0.632596	0.939189	1.161082	
9	.11	0.653633	0.994420	1.129663	
10,11	01 10	0.689907	1.013336	1.154467	1.307453
12	11	0.854545	1.06578	1.190525	1.379619

Table 7.3: Vector Dimension and Codebook Size Assignments for $R = 0.4\text{bpp}$.

	$Q = 4\alpha$	$Q = 8\alpha$	$Q = 16\alpha$	$Q = 32\alpha$	$Q = 64\alpha$
Subband ...00	$k = 1$ $N = 56$	$k = 1$ $N = 64$	$k = 1$ $N = 71$	$k = 2$ $N = 804$	$k = 2$ $N = 1164$
Subband ...01 ...10	$k = 1$ $N = 28$	$k = 1$ $N = 37$	$k = 2$ $N = 221$	$k = 2$ $N = 326$	$k = 2$ $N = 374$
Subband ...11	$k = 1$ $N = 41$	$k = 1$ $N = 53$	$k = 2$ $N = 285$	$k = 2$ $N = 451$	$k = 2$ $N = 590$
Subband ..01 ..10	$k = 1$ $N = 29$	$k = 2$ $N = 123$	$k = 2$ $N = 231$	$k = 2$ $N = 349$	$k = 2$ $N = 414$
Subband ..11	$k = 1$ $N = 11$	$k = 1$ $N = 16$	$k = 2$ $N = 74$	$k = 4$ $N = 538$	$k = 4$ $N = 1902$
Subband .01 .10	$k = 4$ $N = 12$	$k = 4$ $N = 18$	$k = 4$ $N = 29$	$k = 4$ $N = 35$	$k = 4$ $N = 28$
Subband .11	$k = -$ $N = -$	$k = -$ $N = -$	$k = -$ $N = -$	$k = -$ $N = -$	$k = -$ $N = -$
Subband 01 10	$k = -$ $N = -$	$k = -$ $N = -$	$k = -$ $N = -$	$k = -$ $N = -$	$k = -$ $N = -$
Subband 11	$k = -$ $N = -$	$k = -$ $N = -$	$k = -$ $N = -$	$k = -$ $N = -$	$k = -$ $N = -$

Table 7.4: Vector Dimension and Codebook Size Assignments for $R = 0.5\text{bpp}$.

	$Q = 4\alpha$	$Q = 8\alpha$	$Q = 16\alpha$	$Q = 32\alpha$	$Q = 64\alpha$
Subband ...00	$k = 1$ $N = 65$	$k = 1$ $N = 68$	$k = 1$ $N = 86$	$k = 1$ $N = 93$	$k = 2$ $N = 1246$
Subband ...01 ...10	$k = 1$ $N = 35$	$k = 1$ $N = 41$	$k = 1$ $N = 59$	$k = 2$ $N = 406$	$k = 2$ $N = 593$
Subband ...11	$k = 1$ $N = 53$	$k = 1$ $N = 59$	$k = 1$ $N = 88$	$k = 2$ $N = 527$	$k = 2$ $N = 823$
Subband ..01 ..10	$k = 1$ $N = 39$	$k = 2$ $N = 148$	$k = 2$ $N = 246$	$k = 2$ $N = 431$	$k = 2$ $N = 647$
Subband ..11	$k = 1$ $N = 14$	$k = 1$ $N = 17$	$k = 1$ $N = 25$	$k = 2$ $N = 143$	$k = 4$ $N = 1108$
Subband .01 .10	$k = 2$ $N = 8$	$k = 2$ $N = 11$	$k = 4$ $N = 46$	$k = 4$ $N = 77$	$k = 4$ $N = 102$
Subband .11	$k = -$ $N = -$	$k = 4$ $N = 2$	$k = 4$ $N = 5$	$k = 4$ $N = 5$	$k = 4$ $N = 4$
Subband 01 10	$k = -$ $N = -$	$k = -$ $N = -$	$k = -$ $N = -$	$k = -$ $N = -$	$k = -$ $N = -$
Subband 11	$k = -$ $N = -$	$k = -$ $N = -$	$k = -$ $N = -$	$k = -$ $N = -$	$k = -$ $N = -$

Figure 7.2: Average PSNR of Eight Fingerprint Rasterfiles for Various Combinations of R and Q .

Figure 7.2 depicts distortion-rate measurements obtained by compressing an ensemble of five images for various cases of R and Q . In this experiment, no entropy coding of the source symbols was performed, and hence, the bit-rate for each case is equal to the design complexity. We found little improvement when increasing Q over 64. Introduction of the entropy coder can cause the compression obtained by WVQ compression to vary dramatically, depending on the characteristics of the input image; images containing large blank regions in the card background are more compressible. Table 7.5 shows the actual compression ratios achieved for a set of five images when the zero run-length Huffman coding discussed in Section 5 was used to code the codevector indices. Thus, a particular coder should be thought of as compressing full-sized images to a given quality, not by a certain compression ratio.

Figure 7.5 is the image shown in Figure 7.3 compressed by the coder designed for the case of $R = 0.5\text{bpp}$ and $Q = 32\alpha$ as listed in column 4 of Table 7.4. The compression ratio achieved, after the zero run-length Huffman coding of the source symbols, was 21.9 : 1.

Table 7.5: Compression Ratios Achieved for Various Fingerprint Raster Files with Coder Designed for $R = \frac{1}{2}bpp$ and $Q = 512\alpha$.

Image	Compression Ratio
103750.1m	21.0
107879.2d	27.5
114439.3d	24.6
155309.2m	19.2
155498.9m	27.6
582562.8d	31.4
628663.10d	28.3
755946.7m	24.1

8 SOFTWARE DOCUMENTATION.

This section discusses the Los Alamos Fingerprint Compression (LAFPC) software, which consists of a pair of programs implementing the WVQ image compression algorithm. LAFPC uses the four-level octave-scaled wavelet decomposition depicted in Figure 7.4; efforts are under way to incorporate more complex transforms, as discussed in Section 2. The source code is written in the K&R version of the C programming language [9] and is accompanied by a “make file,” which will compile the code on a UNIX system. LAFPC is currently in its third version. In LAFPC 1.0 the VQ encoding search procedure was accomplished with the patented “A-tree” algorithm, which, in version 2.0, was replaced by the public-domain k-d tree search. The differences between LAFPC 2.0 and LAFPC 3.0 are in the entropy coding of the codevector indices: version 2.0 uses straight Huffman coding, while version 3.0 uses the combined Huffman and zero run-length coding.

The LAFPC software reads an auxiliary file that describes the system parameters; the relevant data for each subband are the height and width of the codevector blocks, the number of codevectors, the codebook entries, the Huffman coder table, and the threshold value. These data are packed into a file that, for lack of a better name, is referred to as an “X-file.”

LAFPC 3.0 contains two programs. The first, `compress3`, inputs a fingerprint raster file and outputs the coded image. The second program, `decompress3`, accepts a compressed file and expands it for display. The operation of either program requires the auxiliary X-file. The usage of these two programs is as follows.

```
compress3 input_file output_file xfile
decompress3 input_file output_file xfile
```

Example: To compress the fingerprint raster file “foo” using the coder specified by the data in the X-file labeled “X20,” type the following.

```
compress3 foo foo.C X20
```

This compresses “foo” and puts the result in the file “foo.C”. Using the original file name with a “.C” suffix to denote the compressed file is a convention that is used here but is not

necessary. The output file can have any name the user desires. The command to decompress the file is as follows.

```
decompress3 foo.C foo.D X20
```

This creates the file "foo.D," which is the same size as the original "foo." The ".D" suffix denotes a decompressed file—again, any file name will work for the output file. The usage for LAFPC 3.0 is the same as for previous versions, except that the executables are called **compress3** and **decompress3**.

The companion magnetic tape (sent previously) that goes with this report contains a directory with LAFPC 3.0. The directory contains a subdirectory called "src" that contains all of the C source code and the make file. The directory also contains a subdirectory called "xfiles" containing X-files designed for compression ratios of 10, 16 and 20; the encoding complexity in each case was 32α . The convention used for naming the X-files was X#, where # represents the compression ratio that the coder was designed to realize; e.g., the X-file for compressing 20:1 is called "X20."

Remarks.

- The input to **compress3** and output from **decompress3** are raw image data files, i.e., the image is an array of unsigned characters stored by rows. This is the format of the fingerprint images supplied to Los Alamos by the FBI for the purposes of this research project. The algorithms will not operate with GIF files, TIF files, SUN raster files, or any other such image format.
- At present, the input image must be square and must be of a size divisible by 16, e.g., 512×512 , 768×768 , 160×160 .
- **compress3** and **decompress3** do not know which directory the X-files are in. To avoid typing the path name the user may wish to copy the X-file to the working directory (or create a symbolic link).
- The user must take care to use the same X-file in the compression and decompression processes.
- The k-d tree is not stored in the X-file but rather is generated every time a new codebook is input to the subroutine. Profiling the compression routine showed that the tree-building is a negligible portion of the total execution time.

9 CONCLUSIONS.

This report provides a review of the WVQ algorithm and discusses its adaptation to the compression of fingerprint raster files. Specific implementation details and usage of the LAFPC software, which does the WVQ fingerprint compression, are covered. Experimental results derived from the use of the software are presented to document the algorithm's effectiveness.

There are many avenues for continuing the research presented here. First, the LAFPC software is being updated to employ a more complicated subband decomposition than the

octave-scaled decomposition shown in Figure 7.4; results of FBI investigators in combining an information-minimizing decomposition in conjunction with scalar quantization look very promising. There is also much room for improvement on the entropy coding scheme used for coding the codevector indices; possible avenues for research here include adaptive Huffman coding, arithmetic coding, and the use of higher-order Markov models. The WVQ design algorithm includes an encoding complexity constraint that is based on the full-search procedure, while in practice the coder is implemented with a search routine of greatly reduced complexity. It would be more practical to impose a constraint on codebook memory to limit the resources required by the VQ. The threshold values in Table 5.1 were chosen empirically; a more careful selection certainly could be obtained, possibly through the use of a model of human vision.

References

- [1] J. N. Bradley, "Subband image coding with vector quantization," Ph.D. dissertation, University of Utah, 1989.
- [2] J. N. Bradley, C. M. Brislawn, "Image compression by vector quantization of multiresolution decompositions," *Proceedings of the CNLS 11th Annual Conference on Experimental Mathematics: Computational Issues in Nonlinear Science*, Los Alamos, New Mexico, May 20-24, 1991.
- [3] A. Cohen, I. C. Daubechies, J.-C. Feauveau, "Biorthogonal bases of compactly supported wavelets," AT&T Bell Laboratories Technical Memorandum 11217-900529-07 TM, 1990.
- [4] R. E. Crochiere, L. R. Rabiner, *Multirate Digital Signal Processing*, Englewood Cliffs, N.J. Prentice Hall, 1983.
- [5] J. H. Friedman, J. L. Bentley, "An algorithm for finding best matches in logarithmic expected time," *ACM Transactions on Mathematical Software*, vol. 3, no.3, pp. 209-226, September 1977.
- [6] A. Gersho, "Asymptotically optimal block quantization," *IEEE Trans. Info. Theory*, vol. 32, pp. 373-380, July 1979.
- [7] P. E. Gill, W. Murray, M.A. Saunders, M. H. Wright, "User's guide for NPSOL (Version 4.0): a Fortran package for nonlinear programming," Technical Report SOL 86-2, Department of Operations Research, Stanford University, January 1986.
- [8] R. M. Gray, "Vector quantization," *IEEE ASSP Magazine*, vol. 1, no. 2, pp. 4-29, April 1984.
- [9] B. W. Kernighan, D. M. Ritchie, *The C Programming Language*, Englewood Cliffs, N.J., Prentice Hall, 1978.
- [10] Y. Linde, A. Buzo, R. M. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications*, vol. COM-28, pp. 84-95, January 1980.

- [11] S. P. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. IT-28, pp. 129-137, March 1982.
- [12] J. Makhoul, S. Roucos, H. Gish, "Vector quantization in speech coding," *Proceedings of the IEEE*, vol. 73, no. 11, pp. 1551-1588, November 1985.
- [13] J. G. Proakis, *Digital Communications*, New York, NY, McGraw-Hill, 1983.
- [14] A. Segall, "Bit allocation and encoding for vector sources," *IEEE Transactions on Information Theory*, vol. IT-22, pp. 162-169, March 1976.
- [15] P. L. Zador, "Asymptotic quantization error of continuous signals and the quantization dimension," *IEEE Transactions on Information Theory*, vol. 28, pp. 139-149, March 1982.

Figure 7.3: Original 768 × 768 Fingerprint Image.

Figure 7.1: Four Level Octave Scaled Wavelet Decomposition of Fingerprint Image.

Figure 7.5: Fingerprint Image (compressed 21.9:1).

DATE

FILMED

2 / 16 / 94

END