# A Fast Search Algorithm for Vector Quantization Using Mean Pyramids of Codewords

Chang-Hsing Lee and Ling-Hwei Chen

*Abstract*— **One of the most serious problems for vector quantization, especially for high dimensional vectors, is the high computational complexity of searching for the closest codeword in the codebook design and encoding phases. Although quantizing high dimensional vectors rather than low dimensional vectors results in better performance, the computation time needed for vector quantization grows exponentially with the vector dimension. This makes high dimensional vectors unsuitable for vector quantization. To overcome this problem, a fast search algorithm, under the assumption that the distortion is measured by the squared Euclidean distance, will be proposed. Using the mean pyramids of codewords, the algorithm can reject many codewords that are impossible matches and hence save a great deal of computation time. The algorithm is efficient for high dimensional codeword searches. Experimental results confirm the effectiveness of the proposed method.**

## I. Introduction

Vector quantization (VQ) is a very efficient approach to low-bit-rate image compression [1], [2]. In VQ, the images to be encoded are first decomposed into vectors (i.e., blocks) and then sequentially encoded vector by vector. Each vector is compared with the codewords in the codebook to find the best matching codeword. The key aspect of VQ is to design a good codebook, $Y=\{y_i \mid i=1, 2, ..., N\}$, which contains the most representative codewords and will be used by the encoder and the decoder. In the encoding process, the encoder designs a mapping Q and assigns an index $i$ to each $k$-dimensional input vector $\mathbf{x}=(x_1, x_2, ..., x_k)$, with $Q(\mathbf{x})=y_i=(y_{i1}, y_{i2}, ..., y_{ik})$. In this paper, we will only consider the mapping Q, which is designed to map $\mathbf{x}$ to $y_i$ with $y_i$ satisfying the following condition:

$$d^2(\mathbf{x}, y_i) = \min_j d^2(\mathbf{x}, y_j), \text{ for } j = 1, 2, ..., N, \quad (1)$$

where $d^2(\mathbf{x}, y_j)$ is the distortion of representing the input vector $\mathbf{x}$ by the codeword $y_j$, and is measured by the squared Euclidean distance, i.e.,

$$d^2(\mathbf{x}, y_j) = \sum_{n=1}^{k} (x_n - y_{jn})^2. \quad (2)$$

The decoder has the same codebook as the encoder. In the decoding process, for each index $i$, the decoder merely performs a simple table look-up operation to obtain $y_i$ and then uses $y_i$ to reconstruct the input vector $\mathbf{x}$. Compression is achieved by transmitting or storing the index of a codeword rather than the codeword itself.

From the above description, we see that the compression ratio of VQ is determined by the codebook size and the dimension of the input vectors, and the distortion is dependent on the codebook size and the selection of codewords. Hence, a good codebook design is the main task of VQ. Many algorithms for optimal codebook design have been proposed [3]-[7]. Among these, the most popular was developed by Linde, Buzo and Gray [3] and is referred to as the LBG algorithm. This algorithm is basically an iterative process to minimize the overall distortion of representing the training vectors by their corresponding codewords. Since a full codebook search is needed to find the closest codeword for each training vector, the algorithm is time consuming, especially for high dimensional vectors. To reduce the computation time needed for such an exhaustive search through the codebook, many fast algorithms have been proposed [8]-[21]. Many of them, however, achieve the goal of decreasing the search time at the expense of the coding quality. In this paper, we will propose a search method, based on the mean pyramids of codewords, to speed up the closest codeword search process. This method has the same coding quality as the LBG algorithm and is effective for high dimensional vectors.

In the next section, we will describe the mean pyramid data structure. Section 3 will introduce the proposed approach. Experimental results will be presented in Section 4 to show the effectiveness of the proposed method. Finally, a short conclusion is given in Section 5.

## II. Mean Pyramid Data Structure

Image pyramid data structure was originally developed for image coding by Burt and Adelson [22]. In this data structure, an image is represented hierarchically, with each level corresponding to a reduced-resolution approximation. Given an image $X_n$ of size $2^n \times 2^n$, a pyramid of $X_n$ (see Fig. 1) can be defined as a sequence of matrices $\{X_0, ..., X_{m-1}, X_m, X_{m+1}, ...,$

Paper approved by Barry G. Haskell, the Editor for Image Communications Systems of the IEEE Communications Society. Manuscript received: October 19, 1993; revised June 9, 1994 and October 21, 1994. This work was supported in part by the National Science Council of R. O. C. under contract NSC-83-0404-E009-117.

The authors are with the Department of Computer and Information Science, National Chiao Tung University, 1001 Ta Hsueh Road, Hsinchu, Taiwan 30050, Republic of China.

IEEE Log Number 9411636.

0090-6778/95$4.00 © 1995 IEEE

$X_n\}$ with $X_{m-1}$ having size $2^{m-1}\times 2^{m-1}$ and being a reduced-resolution version of $X_m$. Note that $X_0$ has only one pixel. A pyramid data structure can be formed by successively operating over 2×2 neighboring pixels on the higher levels. That is, the value of a pixel $X_{m-1}(i, j)$ on level m-1 can be obtained from the values of the corresponding 2×2 neighboring pixels $X_m(2i-1, 2j-1)$, $X_m(2i-1, 2j)$, $X_m(2i, 2j-1)$, and $X_m(2i, 2j)$ on level m. In other words, $X_{m-1}(i, j)$ can be obtained by $X_{m-1}(i, j) = f(X_m(2i-1, 2j-1), X_m(2i-1, 2j), X_m(2i, 2j-1), X_m(2i, 2j))$, where $f$ is an operating function.

There are many different types of image pyramids [23]. The simplest pyramid data structure is the mean pyramid, which is formed by successively averaging over the corresponding 2×2 neighboring pixels, i.e., the operating function $f$ is an averaging function. The bottom level is the original image and the top level is the mean value of the image. In the next section, we will present a closest codeword search algorithm based on the mean pyramids of codewords.

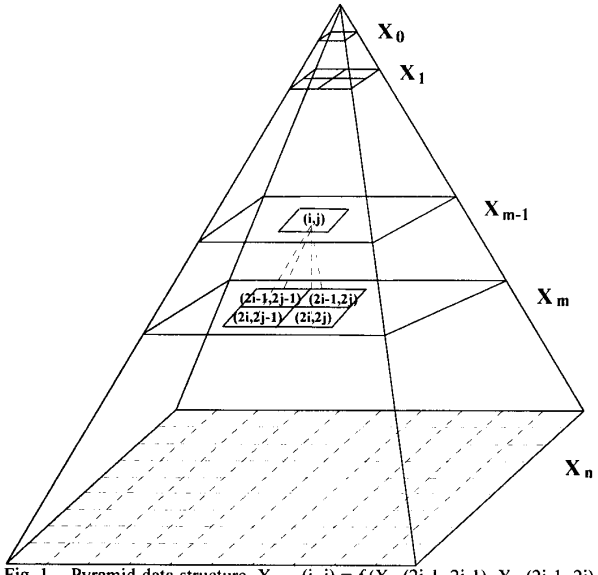


Fig. 1 Pyramid data structure. $X_{m-1}(i, j) = f(X_m(2i-1, 2j-1), X_m(2i-1, 2j), X_m(2i, 2j-1), X_m(2i, 2j))$, where $f$ is an operating function.

## III. The Proposed Algorithm

In this section, we will present the proposed algorithm for the closest codeword search. Based on the mean pyramids of codewords, the algorithm can speed up the search step. Before describing the algorithm, we will give some definitions and a lemma.

Let $\mathbf{x}=(x_1, x_2, \ldots, x_k)$ be a $k$-dimensional vector and $y_i=(y_{i1}, y_{i2}, \ldots, y_{ik})$ be a codeword. Define the mean values of $\mathbf{x}$ and $y_i$ as

$$m_x = \frac{1}{k}\sum_{j=1}^{k} x_j, \tag{3}$$

and

$$m_{yi} = \frac{1}{k}\sum_{j=1}^{k} y_{ij}. \tag{4}$$

And define the distortion, $d^2(\mathbf{x}, y_i)$, of representing $\mathbf{x}$ by $y_i$ as the squared Euclidean distance between $\mathbf{x}$ and $y_i$, that is,

$$d^2(\mathbf{x}, y_i) = \sum_{j=1}^{k} (x_j - y_{ij})^2.$$

**Lemma 1** $d^2(\mathbf{x}, y_i) \geq k(m_x - m_{yi})^2$.

**Proof:** Let $a_j = x_j - y_{ij}$, and $b = m_x - m_{yi}$. Then we have

$$d^2(\mathbf{x}, y_i) = \sum_{j=1}^{k} a_j^2,$$

and

$$\sum_{j=1}^{k} a_j = kb. \tag{5}$$

From (5), we can easily get

$$0 \leq \sum_{j=1}^{k} (a_j - b)^2 = \sum_{j=1}^{k} a_j^2 - kb^2,$$

i.e., $d^2(\mathbf{x}, y_i) = \sum_{j=1}^{k} a_j^2 \geq kb^2 = k(m_x - m_{yi})^2$.

This completes the proof.

Assume the vector dimension $k=2^n\times 2^n$. Then for each vector $\mathbf{x}$ and each codeword $y_i$, we can express them as two $2^n \times 2^n$ vectors $VCT_x$ and $VCT_{yi}$. Two mean pyramids $\{X_0, X_1, \ldots, X_n=VCT_x\}$ and $\{Y_{i0}, Y_{i1}, \ldots, Y_{in}=VCT_{yi}\}$ for $VCT_x$ and $VCT_{yi}$, respectively, can then be constructed. Let $d_m^2(\mathbf{x}, y_i)$ represent the squared Euclidean distance between $X_m$ and $Y_{im}$, i.e.,

$$d_m^2(\mathbf{x}, y_i) = \sum_{j=1}^{2^m}\sum_{h=1}^{2^m} [X_m(j, h) - Y_{im}(j, h)]^2,$$

where $X_m(j, h)$ and $Y_{im}(j, h)$ represent the values of the (j, h)-th pixels on $X_m$ and $Y_{im}$, respectively. Thus, on the top level, $d_0^2(\mathbf{x}, y_i) = (m_x - m_{yi})^2$. From the above definitions and Lemma 1, we have the following theorem.

**Theorem 1** $d^2(\mathbf{x}, y_i) \geq 4\, d_{n-1}^2(\mathbf{x}, y_i) \geq 4^2\, d_{n-2}^2(\mathbf{x}, y_i) \geq \ldots \geq 4^n\, d_0^2(\mathbf{x}, y_i)$. (6)

**Proof:** Since $d_{m+1}^2(\mathbf{x}, y_i)$

$$= \sum_{a=1}^{2^{m+1}} \sum_{b=1}^{2^{m+1}} [X_{m+1}(a, b) - Y_{i(m+1)}(a, b)]^2$$

$$= \sum_{j=1}^{2^m} \sum_{h=1}^{2^m} \{ [X_{m+1}(2j-1, 2h-1) - Y_{i(m+1)}(2j-1, 2h-1)]^2 + [X_{m+1}(2j-1, 2h) - Y_{i(m+1)}(2j-1, 2h)]^2 + [X_{m+1}(2j, 2h-1) - Y_{i(m+1)}(2j, 2h-1)]^2 + [X_{m+1}(2j, 2h) - Y_{i(m+1)}(2j, 2h)]^2 \}.$$

From Lemma 1 and the definition of mean pyramid, for any m, $0 \le m < n$, we have

$$d^2_{m+1}(\mathbf{x}, y_i) \ge \sum_{j=1}^{2^m} \sum_{h=1}^{2^m} 4[X_m(j, h) - Y_{im}(j, h)]^2$$

$$= 4\, d^2_m(\mathbf{x}, y_i).$$

Since $d^2(\mathbf{x}, y_i) = d^2_n(\mathbf{x}, y_i)$, we can easily get

$$d^2(\mathbf{x}, y_i) \ge 4\, d^2_{n-1}(\mathbf{x}, y_i) \ge 4^2\, d^2_{n-2}(\mathbf{x}, y_i) \ge \ldots \ge 4^n\, d^2_0(\mathbf{x}, y_i).$$

With the above theorem in hand, we will begin describing the proposed algorithm. For a training vector **x**, the mean pyramid of **x** is first established and the codeword $y_p$ with the minimum mean difference from **x** is found. Then, the distortion $d^2(\mathbf{x}, y_p)$ is evaluated and the current minimum distortion $d^2_{\min}$ is set to be $d^2(\mathbf{x}, y_p)$. The current closest codeword to **x** is set to be $y_p$. For any other codeword $y_i$, the algorithm starts from the top level of the mean pyramid. It first calculates $d^2_0(\mathbf{x}, y_i)$ and checks if $4^n\, d^2_0(\mathbf{x}, y_i) \ge d^2_{\min}$. If the answer is yes, from Theorem 1, we have $d^2(\mathbf{x}, y_i) \ge d^2_{\min}$, thus codeword $y_i$ will not be the closest one and can be rejected. Otherwise, the squared Euclidean distance, $d^2_1(\mathbf{x}, y_i)$, on the second level is calculated and checked. If $4^{n-1} d^2_1(\mathbf{x}, y_i) \ge d^2_{\min}$, from similar reason as above, codeword $y_i$ can be rejected. If it is not rejected, the third level is tested. This process is repeated until $y_i$ is rejected or the bottom level is reached. If the bottom level is reached, then the distortion $d^2(\mathbf{x}, y_i)$ is calculated and checked. If $d^2(\mathbf{x}, y_i) < d^2_{\min}$, the current minimum distortion $d^2_{\min}$ is replaced by $d^2(\mathbf{x}, y_i)$ and the current closest codeword to **x** is set to be $y_i$.

In the top level check of the mean pyramid, the algorithm will reject those codewords with mean values different from that of **x**. Moreover, the algorithm will cause many codewords to be rejected when the intermediate levels of their mean pyramids are checked. This will prevent codewords that have mean values similar to $m_x$ but are very different from **x** from being considered as candidates for the closest codeword to **x**. Hence, the proposed algorithm is a "coarse to fine" technique, which will reject many codewords without evaluating their distortion in order to speed up the search process. This approach is effective for high dimensional vectors. Since evaluating the squared Euclidean distance between two high dimensional vectors would be an immense task, rejecting many codewords before the actual squared Euclidean distances are evaluated can save a great deal of time.

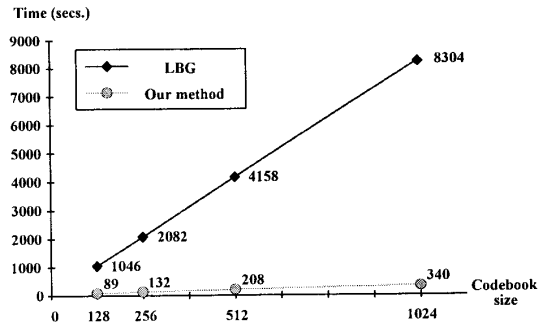


Fig. 2 The execution time for codebook design with block size 4×4.

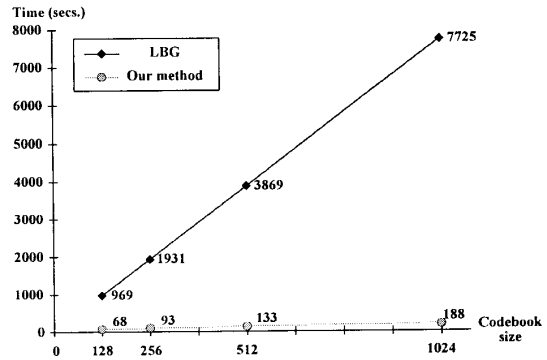


Fig. 3 The execution time for codebook design with block size 8×8.

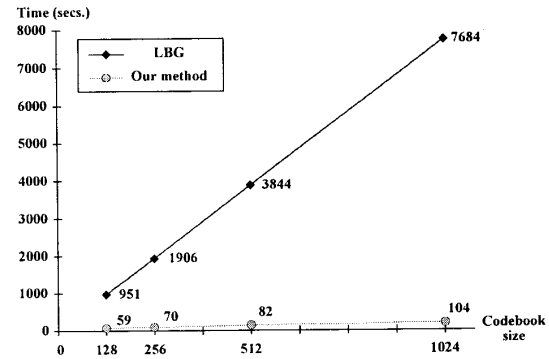


Fig. 4 The execution time for codebook design with block size 16×16.

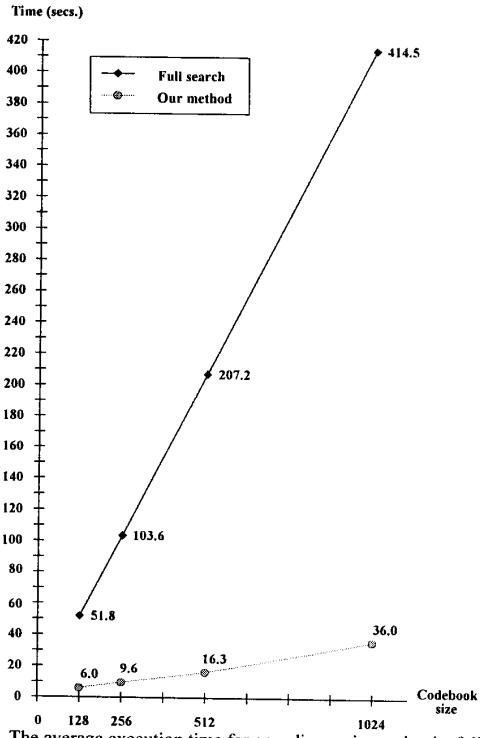


Fig. 5 The average execution time for encoding an image by the full search method and the proposed algorithm with block size 4×4.

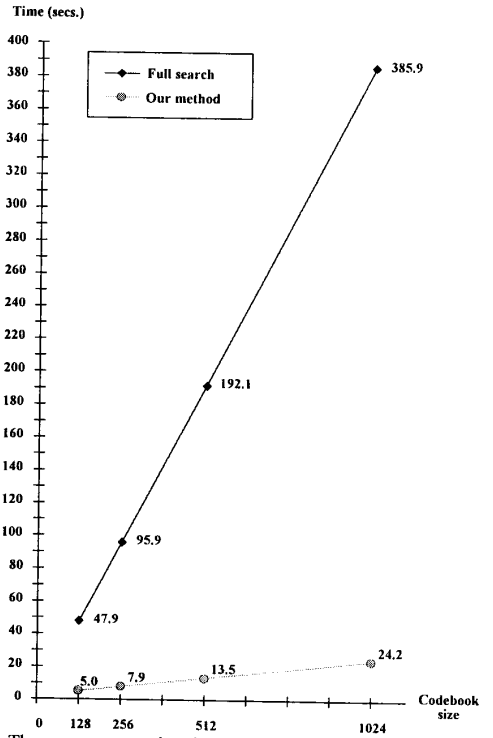


Fig. 6 The average execution time for encoding an image by the full search method and the proposed algorithm with block size 8×8.

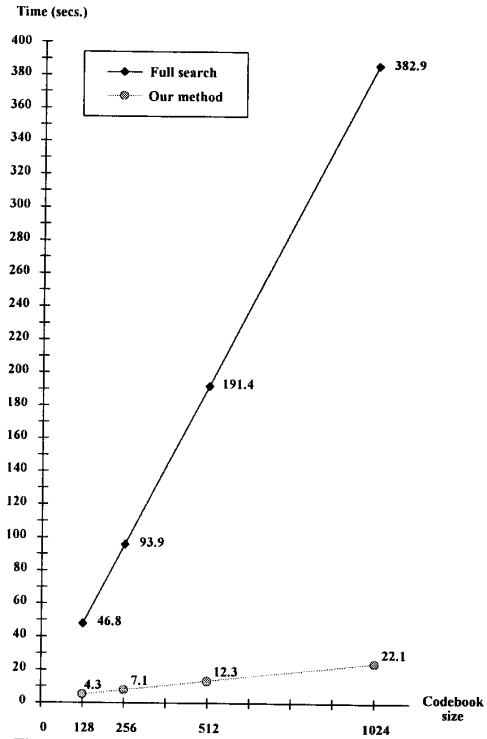


Fig. 7 The average execution time for encoding an image by the full search method and the proposed algorithm with block size 16×16.

The proposed algorithm can also be applied to encoder to find the closest codeword for each input vector. The search process is similar to that of the codebook design algorithm described above.

## IV. Simulation Results

The efficiency of the proposed algorithm was examined by means of experiments performed on a Sun SPARC-station-IPC using several $512 \times 512$ monochrome images with 256 gray levels. The performance is evaluated in codebook design as well as image encoding in terms of execution time and the number of distortion calculations.

Figures 2-4 show the execution time required for codebook design with block sizes $4 \times 4$, $8 \times 8$, and $16 \times 16$, respectively. The simulations use the Lena image as a codebook design. From these figures, we see that the proposed algorithm dramatically reduces the execution time required by the LBG algorithm. Figures 5-7 show the average execution time needed to encode an image. In the simulations, the Lena image is used to design a codebook, and the resulting codebook is then used to encode the four images (Lena, Peppers, Jet, and Baboon). From these figures, we can see that a great deal of execution time has been reduced.

Table 1 compares the average number of distortion calculations and overhead required for each vector in codebook design with block sizes $4 \times 4$, $8 \times 8$, and $16 \times 16$,

respectively. Table 2 compares the average number of distortion calculations needed for each vector in image encoding. The overhead of intermediate level checks in the mean pyramid is also included in this table. From these two tables, we can see that only a small number of distortion calculations has to be calculated by the proposed algorithm. This also means that a lot of codewords have been rejected in the intermediate level checks before they reach the bottom level of the mean pyramid. And the overhead is very small.

TABLE I
THE AVERAGE NUMBER OF DISTORTION CALCULATIONS FOR EACH VECTOR IN CODEBOOK DESIGN WITH BLOCK SIZE $4\times4$, $8\times8$, AND $16\times16$. THE VALUES IN PARENTHESES DENOTE THE OVERHEAD.

| Method | Codebook Size | Block size | | |
|---|---|---|---|---|
| | | 4x4 | 8x8 | 16x16 |
| LBG | 128 | 128 (0) | | |
| | 256 | 256 (0) | | |
| | 512 | 512 (0) | | |
| | 1024 | 1024 (0) | | |
| Our method | 128 | 3.4 (2.6) | 3.2 (2.1) | 2.9 (1.5) |
| | 256 | 4.8 (4.8) | 3.9 (3.4) | 3.2 (1.9) |
| | 512 | 6.7 (8.7) | 4.8 (5.5) | 3.2 (2.1) |
| | 1024 | 9.6 (15.7) | 5.4 (7.8) | 3.2 (2.3) |

TABLE II
THE AVERAGE NUMBER OF DISTORTION CALCULATIONS (INCLUDING OVERHEAD) FOR EACH VECTOR IN IMAGE ENCODING WITH BLOCK SIZE $4\times4$, $8\times8$, AND $16\times16$.

| Method | Codebook Size | Block size | | |
|---|---|---|---|---|
| | | 4x4 | 8x8 | 16x16 |
| Full search | 128 | 128 | | |
| | 256 | 256 | | |
| | 512 | 512 | | |
| | 1024 | 1024 | | |
| Our method | 128 | 9.8 | 9.3 | 8.7 |
| | 256 | 17.0 | 15.5 | 14.1 |
| | 512 | 29.9 | 26.5 | 23.7 |
| | 1024 | 53.6 | 46.1 | 41.3 |

## V. CONCLUSIONS

In this paper, we have presented a fast search algorithm for vector quantization based on the assumptions that the distortion is measured by the squared Euclidean distance and the vector dimension is $2^n\times2^n$. The algorithm is a "coarse to fine" technique, it uses the mean pyramids of codewords to reject many unmatched codewords, thus dramatically speeding up the search process in VQ codebook design and image encoding. The performance of the algorithm is evaluated in both codebook design as well as image encoding. Simulation results show that the proposed algorithm greatly reduces the execution time needed for both of codebook design and image encoding. Moreover, the algorithm is effective for large block image coding. The problem with the proposed algorithm is the storage requirement. Since the proposed algorithm needs to calculate and store the mean pyramid of each codeword, it needs extra 1/3 storage than the LBG algorithm.

## REFERENCES


[1] R. M. Gray, "Vector quantization," *IEEE ASSP Magazine*, vol. 1, pp. 4-9, Apr. 1984.

[2] N. M. Nasrabadi and R. A. King, "Image coding using vector quantization: A review," *IEEE Trans. Commun.*, vol. COM-36, no. 8, pp. 957-971, Aug. 1988.

[3] Y. Linde, A. Buzo, and R. M. Gray "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. COM-28, no. 1, pp. 84-95, Jan. 1980.

[4] W. H. Equitz, "A new vector quantization clustering algorithm," *IEEE Trans. Acoust., Speech, and Signal Processing*, vol. 37, no. 10, pp. 1568-1575, Oct. 1989.

[5] B. Marangelli, "A vector quantizer with minimum visible distortion," *IEEE Trans. Signal Processing*, vol. 39, no. 12, pp. 2718-2721, Dec. 1991.

[6] K. Zeger, J. Vaisey, and A. Gersho, "Globally optimal vector quantizer design by stochastic relaxation," *IEEE Trans. Signal Processing*, vol. 40, no. 2, pp. 310-322, Feb. 1992.

[7] K. Rose, E. Gurewitz, and G. C. Fox, "Vector quantization by deterministic annealing," *IEEE Trans. Inform. Theory*, vol. IT-38, no. 4, pp. 1249-1257, Jul. 1992.

[8] C. D. Bei and R. M. Gray "An improvement of the minimum distortion encoding algorithm for vector quantization," *IEEE Trans. Commun.*, vol. COM-33, no. 10, pp. 1132-1133, Oct. 1985.

[9] K. K. Paliwal and V. Ramasubramanian, "Effect of ordering the codebook on the efficiency of the partial distance search algorithm for vector quantization," *IEEE Trans. Commun.*, vol. COM-37, no. 5, pp. 538-540, May. 1989.

[10] C. H. Hsieh, P. C. Lu, and J. C. Chang "Fast codebook generation algorithm for vector quantization of images," *Pattern Recognition Letters*, vol. 12, pp. 605-609, 1991.

[11] D. Y. Cheng, A. Gersho, B. Ramamurthi, and Y. Shoham, "Fast search algorithms for vector quantization and pattern matching," in *Proc. IEEE ICASSP*, 1984, pp. 9.11.1-9.11.4.

[12] D. Y. Cheng and A. Gersho "A fast codebook search algorithm for nearest-neighbor pattern matching," in *Proc. IEEE ICASSP*, 1986, pp. 265-268.

[13] A. Lowry, S. Hossain, and W. Millar, "Binary search trees for vector quantization," in *Proc. IEEE ICASSP*, 1987, pp. 2205-2208.

[14] V. Ramasubramanian and K. K. Paliwal, "An optimized k-d tree algorithm for fast vector quantization of speech," in *Proc. European Signal Processing Conf.*, 1988, pp. 875-878.

[15] V. Ramasubramanian and K. K. Paliwal, "Fast k-dimensional tree algorithms for nearest neighbor search with application to vector quantization encoding," *IEEE Trans. Signal Processing*, vol. 40, no.3, pp. 518-531, Mar. 1992.

[16] M. R. Soleymani and S. D. Morgera, "A high-speed search algorithm for vector quantization," in *Proc. IEEE ICASSP*, 1987, pp. 45.6.1-3.

[17] V. Ramasubramanian and K. K. Paliwal, "An efficient approximation-elimination algorithm for fast nearest-neighbor search based on a spherical distance coordinate formulation," *Pattern Recognition Letters*, vol. 13, pp. 471-480, 1992.

[18] E. Vidal, "An algorithm for finding nearest neighbors in (approximately) constant average time complexity," *Pattern Recognition Letters*, vol. 4, pp. 145-157, 1986.

[19] M. T. Orchard, "A fast nearest-neighbor search algorithm," in *Proc. IEEE ICASSP*, 1991, pp. 2297-2300.

[20] C. M. Huang, Q. Bi, G. S. Stiles, and R. W. Harris, "Fast full search equivalent encoding algorithms for image compression using vector

quantization," *IEEE Trans. Image Processing*, vol. 1, no. 3, pp. 413-416, Jul. 1992.
[21] L. Guan and M. Kamel, "Equal-average hyperplane partitioning method for vector quantization of image data," *Pattern Recognition Letters*, vol. 13, pp. 693-699, 1992.
[22] P. J. Burt and E. Adelson, "The Laplacian pyramid as a compact image code," *IEEE Trans. Commun.*, vol. COM-31, no. 4, pp. 532-540, Apr. 1983.
[23] L. Wang and M. Goldberg, "Reduced-difference pyramid: a data structure for progressive image transmission," *Optical Engineering*, vol. 28, no. 7, pp. 708-716, Jul. 1989.

**Chang-Hsing Lee** was born on July 24, 1968 in Tainan, Taiwan, Republic of China. He received the B.S. degree in Computer and Information Science from National Chiao Tung University in 1991. Since 1992 he has been studying toward the Ph.D. degree and, currently, he is a Ph.D. candidate in the Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan. His main research interests include image compression, pattern recognition, image processing and neural networks.

**Ling-Hwei Chen** received the B.S. degree in mathematics and the M.S. degree in applied mathematics from National Tsing Hua University, Taiwan, in 1975 and 1977, respectively, and the Ph.D. degree in computer engineering from National Chiao Tung University, Taiwan, in 1987. From 1977 to 1979, she worked as a research assistant in the Chung-Shan Institute of Science and Technology, Taiwan. From 1979 to 1981, she worked as a research associate at the Electronic Research and Service Organization, Industry Technology Research Institute, Taiwan. From 1981 to 1983, she worked as an engineer at the Institute if Information Industry, Taiwan. She joined the Department of Computer and Information Science at National Chiao Tung University in 1987 and is currently a professor there. Her current research interests include image processing and pattern recognition.