Pattern Recognition Letters 22 (2001) 373–379

Pattern Recognition Letters

www.elsevier.nl/locate/patrec

An improved search algorithm for vector quantization using mean pyramid structure

Su-Juan Lin, Kuo-Liang Chung [*,1], Lung-Chun Chang

Department of Information Management, Institute of Computer Science and Information Engineering, National Taiwan University of Science and Technology, No. 43, Section 4, Keelung Road, Taipei, Taiwan 10672, ROC

Received 14 January 2000; received in revised form 9 August 2000

Abstract

Vector quantization (VQ) is a well-known data compression technique. In the codebook design phase as well as the encoding phase, given a block represented as a vector, searching the closest codeword in the codebook is a time-consuming task. Based on the mean pyramid structure and the range search approach, an improved search algorithm for VQ is presented in this paper. Conceptually, the proposed algorithm has the bandpass filter effect. Each time, using the derived formula, the search range becomes narrower due to the elimination of some portion of the previous search range. This reduces search times and improves the previous result by Lee and Chen (A fast search algorithm for vector quantization using mean pyramids of codewords. IEEE Trans. Commun. 43(2/3/4), (1995) 1697–1702). Some experimental results demonstrate the computational advantage of the proposed algorithm. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Codebook; Fast search algorithm; Mean pyramid; Vector quantization

1. Introduction

Vector quantization (VQ) is an important technique (Gersho and Gray, 1992; Gray and Neuhoff, 1998) for low-bit-rate image compression. It can be defined as a mapping function Q from a k-dimensional Euclidean space R^k to a finite subset C of R^k:

$$Q : R^k \rightarrow C,$$

where $C = \{c_i \mid i = 1, 2, \dots, N\}$ is the codebook with size N and each possible mapped codeword $c_i = (c_{i1}, c_{i2}, \dots, c_{ik})$ in $C \subset R^k$ is of k-dimension.

The process of VQ can be divided into three phases: (1) codebook generation, (2) encoding, and (3) decoding. Given a large amount of vectors, i.e., blocks, the goal of codebook design is to build up a codebook C which contains the most representative codewords, and then this constructed codebook will be used by both encoder and decoder. Many algorithms for optimal codebook design (Linde et al., 1980; Rose et al., 1992; Zeger et al., 1992) have been proposed. Among them, the most popular one was developed by Linde et al. (1980) and is referred to as the LBG algorithm. This research focuses on the

[*] Corresponding author. Tel.: +886-2-2737-6771; fax: +886-2-2737-6777.
E-mail address: klchung@cs.ntust.edu.tw (K.-L. Chung).
[1] Supported by NSC88-2213-E011-005/006.

encoding phase and then is applied to speed up the LBG algorithm.

In the encoding phase, the encoder first divides the image into many square blocks (or vectors) and each vector is with dimension $k(=\sqrt{k}\times\sqrt{k})$. Next, the encoder wants to design a mapping function Q such that given a vector $x=(x_1,x_2,\dots,x_k)$, the squared Euclidean distance between x and the mapped vector $Q(x)=c_i$ is smallest:

$$d^2(x,c_i)=\min_j\sum_{n=1}^{k}(x_n-c_{jn})^2.$$

That is, the corresponding distortion is minimal. Then, the vector x is replaced by the index i of c_i. Since the number of bits used for representing the index is always smaller than that of the vector x, the encoded image is thus compressed. In the decoding phase, the decoder has the same codebook as the encoder. The decoder has index i as input and merely performs a simple table lookup operation to obtain the decoded codeword c_i and then uses c_i to reconstruct the input vector x approximately. This low complexity decompression in VQ has good computational advantage in comparison with the other compression techniques requiring extra computation in the decoding phase. Fig. 1 shows the diagram of VQ.

From the above description, we can see that there are two major problems in VQ. One is to generate a representative codebook efficiently. The other is to reduce the search-time complexity in the encoding phase. We have mentioned that the LBG algorithm is the most popular codebook design algorithm. Basically, the LBG algorithm is an iterative process to minimize the overall distortion for representing the training vectors by their representative codewords. The LBG algorithm uses a full codebook search to find the closest codeword for each training vector in order to update the current codeword. A full codebook search is also used to the encoding phase. It is, however, time-consuming. For each vector x, a full search requires N distortion calculations, i.e., squared Euclidean distance calculations, to find the codeword in C that is closest to x. These distortion calculations need Nk multiplications, $N(2k-1)$ additions, and $N-1$ comparisons. Here, we assume that the time required for computing one addition is equal to that of one subtraction. In order to reduce the time bound requirement for VQ, many efficient search algorithms (Bei and Gray, 1985; Chang and Lin, 1997; Gray and Linde, 1982; Jo and Kaimal, 1999; Lee and Chen, 1995; Lee and Chen, 1995; Ra and Kim, 1993; Torres and Huguet, 1994) have been proposed. Among them, an efficient VQ search algorithm based on the mean pyramid structure was presented by Lee and Chen (1995).

Following the same pyramid structure as in (Lee and Chen, 1995) and the range search

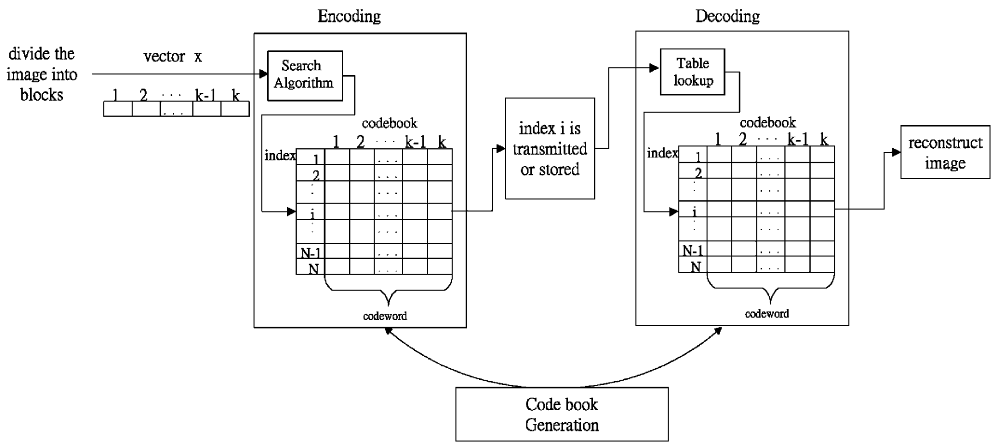

Fig. 1. VQ diagram.

approach, an improved search algorithm for VQ is presented in this paper. Each time, using the derived formula, the search range in the current iteration becomes narrower due to the elimination of some portion of the search range in the previous iteration. This reduces search times and improves the previous result by Lee and Chen (1995). The extra cost in the proposed search algorithm is only a simple auxiliary data structure. Some real images are used to carry out the experiments. Experimental results demonstrate the computational advantage of the proposed algorithm.

The rest of this paper is organized as follows. In Section 2, we introduce the mean pyramids structure and the search method by Lee and Chen (1995). Our proposed search algorithm is presented in Section 3. In Section 4, some experimental results are presented to show the advantage of the proposed algorithm. Finally, some conclusions are presented in Section 5.

2. The work of Lee and Chen

Pyramid data structure was originally developed for image coding by Burt and Adelson (1983). That structure is also suited for progressive image transmission. The mean pyramid structure is shown in Fig. 2. Let the root of the mean pyramid be at level 0. Since each codeword is of size $k = \sqrt{k} \times \sqrt{k}$, the depth of one mean pyramid is $l (= \log_4 k)$. The size of the image I_i at level i is $2^i \times 2^i$ for $0 \leqslant i \leqslant l$. In I_{i-1}, the pixel at position $([\frac{x+1}{2}], [\frac{y+1}{2}])$ is the mean of the related four pixels in I_i by computing

$$I_{i-1}\left(\left[\frac{x+1}{2}\right], \left[\frac{y+1}{2}\right]\right) = [(I_i(x,y) + I_i(x,y+1) + I_i(x+1,y) + I_i(x+1,y+1))/4]$$

for $x, y = 1, 3, \ldots, 2^i - 1$, where $[t]$ denotes the truncation of $t + 0.5$.

Suppose we have constructed the two mean pyramids for the vector x and the vector c_i, respectively. Based on the two mean pyramids, (Lee and Chen, 1995) proposed an efficient search algorithm for finding the closest codeword in the codebook. They first derived the following inequalities:

$$d_l^2(x,c_i) \geqslant 4d_{l-1}^2(x,c_i) \geqslant 4^2 d_{l-2}^2(x,c_i) \geqslant \cdots \geqslant 4^l d_0^2(x,c_i), \quad (1)$$

where $d_j^2(x, c_i)$ denotes the squared Euclidean distance between the reduced vector x and the reduced vector c_i at level j in the two mean pyramids. For example, at root level, the reduced vector x (c_i) denotes the mean value of the original vector x (c_i). That is,

$$d_0^2(x, c_i) = \left(\left(\frac{1}{k}\sum_{i=1}^{k} x_i\right) - \left(\frac{1}{k}\sum_{i=1}^{k} c_i\right)\right)^2.$$

At bottom level, we have $d_l^2(x, c_i) = d^2(x, c_i)$.

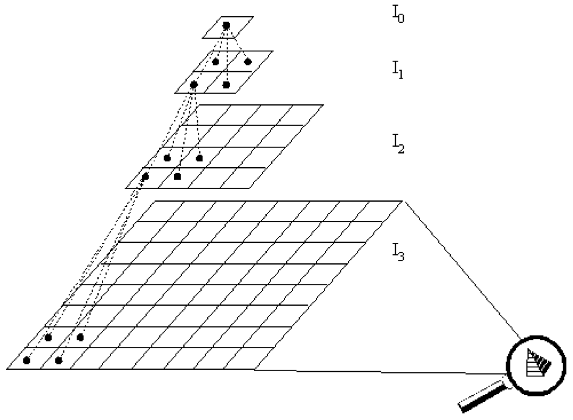

Fig. 2. Mean pyramid structure.

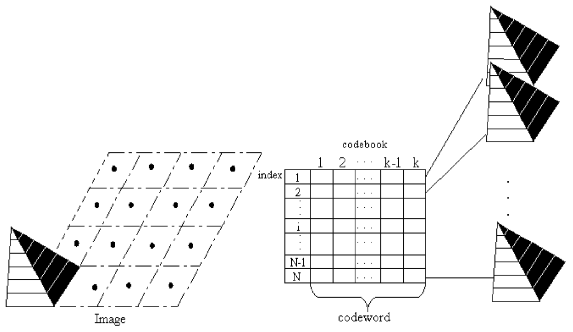

Fig. 3. Mean pyramids of codewords.

Their algorithm consists of the following the three steps:

Step 1. Construct the mean pyramids of codewords. If the codebook size is N, N mean pyramids (see Fig. 3) are constructed. That is, each codeword in the codebook is associated with one mean pyramid.

Step 2. Select a codeword c_p in the codebook to be the current closest codeword to the input vector x.

2.1 For each input vector x, the mean pyramid of x is constructed and the codeword c_p with the minimum mean difference from x is found.

2.2 The distortion $d^2(x, c_p)$ is calculated and the current minimum distortion $d^2_{\min}$ is set to be $d^2(x, c_p)$.

Step 3. For any other codeword c_i in the codebook, it starts from the top level, i.e., level 0, of the mean pyramid and goes downward to the bottom level, i.e., level l, if necessary.

3.1 Calculate $d_0^2(x, c_i)$. Owing to the inequalities (see Eq. (1)), if $4^l d_0^2(x, c_i) \geqslant d^2_{\min}$, then $4^{l-j} d_j^2(x, c_i) \geqslant d^2_{\min}$ will hold for $1 \leqslant j \leqslant l$. Thus, codeword c_i will not be the closest one and can be rejected. Otherwise, the squared Euclidean distance at level 1 is calculated and checked. Similarly, if $4^{l-1} d_1^2(x, c_i) \geqslant d^2_{\min}$, then codeword c_i can be rejected; otherwise, the squared Euclidean distance at level 2 is tested. This comparison process is repeated until c_i is rejected or the bottom level of the mean pyramid for c_i is reached.

3.2 If the bottom level is reached, $d^2(x, c_i)$ is calculated. Then we check whether $d^2_{\min}$ should be replaced or not. If $d^2_{\min}$ is replaced, then the current closest codeword to x is set to be c_i.

The efficiency of their algorithm (Lee and Chen, 1995) results from rejecting many codewords in the codebook before the actual squared Euclidean distances between them and the given vector x are calculated.

In next section, based on the derived formula, an improved search algorithm is presented to improve the search algorithm by Lee and Chen (1995).

3. The proposed improved search algorithm

In our improved algorithm, an auxiliary data structure (ADS) associated with double links is build up to point to the mean pyramids of codewords. The ADS mainly stores the sorted means of all c_i's for $1 \leqslant i \leqslant N$. It will be used to support the search process.

With the ADS, we can efficiently find codeword c_p that has the minimum mean difference from x, using the well-known binary search method (Cormen et al., 1990). We can also narrow the search range for checking whether the squared Euclidean distance calculations are necessary for other codewords.

We choose such a codeword c_p to be the current closest codeword, and the squared Euclidean distance between x and c_p is calculated. Let $d^2(x, c_p)$ be the squared Euclidean distance between x and c_p. Each codeword in the codebook whose squared Euclidean distance from x is greater than $d^2(x, c_p)$, will not be a candidate for the closest codeword of x. As a result, we want to design an efficient search algorithm to reject these infeasible codewords in the early stages.

In what follows, the value $d^2(x, c_p)$ and Eq. (1) are used to calculate two bounds to narrow the search space of the remaining feasible candidates in the codebook for the closest codeword of x. Suppose the codeword c_i is a candidate for the closest codeword of x, then from Eq. (1), it satisfies the following inequalities:

$$\begin{aligned} d^2(x, c_p) &\geqslant 4 d^2_{l-1}(x, c_i) \geqslant 4^2 d^2_{l-2}(x, c_i) \\ &\geqslant \cdots \geqslant 4^l d_0^2(x, c_i). \end{aligned}$$

Consider level 0 of the two mean pyramids of x and c_i. The mean values of x and c_i, denoted by $mean_x$ and $mean_{c_i}$, respectively, are stored at level 0. From the above inequalities, taking the leftmost term and the rightmost term, we have

$$d^2(x, c_p) \geqslant 4^l (mean_{c_i} - mean_x)^2.$$

The both sides of the above inequality are first divided by 4^l, then we perform the square root operations on both sides. We thus obtain the following lower bound and the upper bound of the range if the codeword c_i is a candidate for the closest codeword of x (Lin et al., 1999):

$$\underbrace{mean_x - \sqrt{\frac{\mathrm{d}^2(x,c_p)}{4^l}}}_{\text{lower bound}} \leqslant mean_{c_i} \leqslant \underbrace{mean_x + \sqrt{\frac{\mathrm{d}^2(x,c_p)}{4^l}}}_{\text{upper bound}}. \tag{2}$$

The result of the two bounds of the range in Eq. (2) is similar to the method in (Jo and Kaimal, 1999), but the derivation is different.

In general, as the candidates for the closest codeword of x, the mean values of these candidates must lie between the lower bound and the upper bound (see Eq. (2)). On the contrary, if the mean values of those codewords are out of the range bounded by the lower bound and the upper bound, they will not be possible candidates for the closest codeword of x. Eq. (2) works like a bandpass filter. Before calculating actual squared Euclidean distances between all the remaining codewords and the vector x, this filter can quickly filter out impossible candidates. According to the experimental results mentioned in Section 4, it is observed that applying Eq. (2) to the top level once, the rejection percentage is 72.61–84.16%. Further, if Eq. (2) is used again at the lower levels for the candidate codewords, the rejection percentage improves by only 1–3%. Therefore, throughout the paper, we only apply Eq. (2) to the top level once.

Since each codeword is associated with a mean pyramid, totally we have N mean pyramids for the N codewords. We then take the N means at the top levels of these N mean pyramids. Further, these N means are sorted and the sorted N mean values are stored in ADS as shown in Fig. 4. In the ADS, each entry in the array has a double-link mechanism (Cormen et al., 1990) to access from the codebook to ADS, or vice verse.

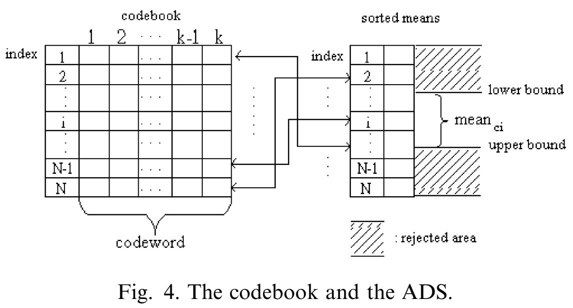

Fig. 4. The codebook and the ADS.

When compared to the memory required to store those N mean pyramids, the memory required for ADS is infinitesimal since it needs only $O(N)$ memory. Based on the proposed ADS and Eq. (2), our formal search algorithm is listed below:

Step 1. Construct the mean pyramids of codewords.

1.1 Sort the N means at the top levels of those N mean pyramids and store the sorted N means in the ADS.

Step 2. For each input vector x, the mean pyramid of x is constructed.

2.1 Perform binary search on array to find the codeword c_p in the ADS with the minimum mean difference from x. The distortion $d^2(x,c_p)$ is evaluated and the current minimum distortion $d^2_{\min}$ is set to be $d^2(x,c_p)$.

2.2 Calculate the lower bound $low_p = mean_x - \sqrt{d^2(x,c_p)/4^l}$ and the upper bound $up_p = mean_x + \sqrt{d^2(x,c_p)/4^l}$. With the help of performing binary search twice, we can locate low_p and up_p. Using the double-link mechanism in the ADS, all the feasible candidates in the codebook for the closest codeword of x can be determined.

Step 3. For any other codeword c_i whose mean, i.e., $mean_{c_i}$, is within the range low_p and up_p, we start from level 1 of the mean pyramid and go downward to the bottom level if necessary. The rest of this step is the same as Step 3 of the previous algorithm described in Section 2.

4. Experimental results

Some real 512×512 gray-level images were used to evaluate the performance of the full search (FS) method, Lee and Chen's search method in (Lee and Chen, 1995), and our proposed method. The three methods were implemented using a Pentium PC with 300 MHz. Two types of experiments are carried out to compare the three methods thoroughly.

In the first type of experiment, we used Lena image as the training set for codebook design. The algorithm of codebook design used is the LBG algorithm (Linde et al., 1980), but the search methods within the LBG algorithm are the FS method, Lee and Chen's method, and our proposed method, respectively. The improvement ratio of the execution time required in the proposed method over the FS method and Lee and Chen's method are denoted by

$$R_1 = \frac{T_{\mathrm{FS}} - T_{\mathrm{ours}}}{T_{\mathrm{FS}}} \times 100\%$$

and

$$R_2 = \frac{T_{[9]} - T_{\mathrm{ours}}}{T_{[9]}} \times 100\%,$$

respectively. Table 1 illustrates the related performance comparison among the three methods, where the time unit is second denoted by 'sec'. Here T_{FS}, $T_{[9]}$, and T_{ours} denote the execution time required in the FS method, Lee and Chen's method, and our proposed method, respectively.

In the second type of experiments, after generating a codebook from Lena image, four images, Lena, F-16, Pepper, and Baboon, were used to calculate the average execution time for encoding each image. The improvement ratio of the average execution time required in the proposed method over the FS method and Lee and Chen's method are denoted by

Table 1
Comparison of execution time for codebook design

Block size	Codebook size	T_{FS} (s)	$T_{[9]}$ (s)	T_{ours} (s)	R_1 (%)	R_2 (%)
2 × 2	128	645	223	95	85.27	57.40
	256	1301	502	191	85.32	61.95
	512	2517	837	270	89.27	67.74
	1024	5340	1805	524	90.19	70.97
4 × 4	128	674	178	93	86.20	47.75
	256	1344	330	160	88.09	51.51
	512	2675	674	284	89.38	57.86
	1024	5479	1225	509	90.70	58.44
8 × 8	128	700	111	92	86.85	17.11
	256	1382	156	133	90.37	14.74
	512	3669	361	303	91.74	16.06
	1024	5443	492	379	93.03	22.96

Table 2
Comparison of average execution time for encoding each image

Block size	Codebook size	T_{FS} (s)	$T_{[9]}$ (s)	T_{ours} (s)	R_1 (%)	R_2 (%)
2 × 2	128	210	91	45	78.57	50.55
	256	425	205	88	79.29	57.07
	512	860	326	137	84.41	57.98
	1024	1747	731	259	85.17	64.57
4 × 4	128	221	72	41	81.44	43.05
	256	441	136	74	83.21	45.58
	512	879	260	133	84.86	48.84
	1024	1768	506	251	85.80	50.39
8 × 8	128	219	41	37	83.15	9.76
	256	444	76	67	84.90	11.84
	512	870	140	128	85.28	8.57
	1024	1776	274	241	86.43	12.04

$$R_1 = \frac{T_{\mathrm{FS}} - T_{\mathrm{ours}}}{T_{\mathrm{FS}}} \times 100\%$$

and

$$R_2 = \frac{T_{[9]} - T_{\mathrm{ours}}}{T_{[9]}} \times 100\%,$$

respectively. Table 2 illustrates the related performance comparison among the three methods. It is observed that our proposed method outperforms the FS method and the method by Lee and Chen. Finally, as can be seen from Tables 1 and 2, the larger the codebook size is, the better the improvement ratio is.

5. Conclusion

The improved search algorithm for VQ has been presented and some experiments have been carried out to confirm the computational advantage of the proposed method when compared to the previous method (Lee and Chen, 1995). In fact, the results of this paper can be applied to motion estimation in video coding (Lee and Chen, 1997).

Acknowledgements

The authors would like to thank the three anonymous referees for their valuable suggestions that lead to the improved presentation of this paper.

References

Bei, C.D., Gray, R.M., 1985. An improvement of the minimum distortion encoding algorithm for vector quantization. IEEE Trans. Commun. 33 (10), 1132–1133.

Burt, P.J., Adelson, E.H., 1983. The Laplacian pyramid as a compact image code. IEEE Trans. Commun. 31 (4), 532–540.

Chang, C.C., Lin, D.C., 1997. An improved VQ codebook search algorithm using principal component analysis. J. Visual Commun. Image Representation 8 (1), 27–37.

Cormen, T.H., Leiserson, C.E., Rivest, R.L., 1990. Introduction to Algorithms. MIT Press, Cambridge, MA.

Gersho, A., Gray, R.M., 1992. Vector Quantization and Signal Compression. Kluwer Academic Publishers, Massachusetts.

Gray, R.M., Linde, Y., 1982. Vector quantization and predictive quantizers for Gauss–Markov sources. IEEE Trans. Commun. 30 (2), 381–389.

Gray, R.M., Neuhoff, D.L., 1998. Quantization. IEEE Trans. Inform. Theory 44 (6), 2325–2383.

Jo, L., Kaimal, M.R., 1999. A fast second-generation encoding algorithm for vector quantization. IEEE Signal Process. Lett. 6 (11), 277–280.

Lee, C.H., Chen, L.H., 1995. A fast search algorithm for vector quantization using mean pyramids of codewords. IEEE Trans. Commun. 43 (2/3/4), 1697–1702.

Lee, C.H., Chen, L.H., 1997. A fast motion estimation algorithm based on the block sum pyramid. IEEE Trans. Image Process. 6 (11), 1587–1591.

Lee, C.H., Chen, L.H., 1995. High-speed closest codeword search algorithm for vector quantization. Signal Process. 43 (3), 323–331.

Lin, S.J., Chung, K.L., Chang, L.C., 1999. A faster search algorithm for vector quantization based on mean pyramids structure. Research report, Department of Information Management, February.

Linde, Y., Buzo, A., Gray, R.M., 1980. An algorithm for vector quantizer design. IEEE Trans. Commun. 28 (1), 84–95.

Ra, S.W., Kim, J.K., 1993. A fast mean-distance-ordered partial codebook search algorithm for image vector quantization. IEEE Trans. Circuits System II: Analog Digital Signal Process. 40 (9), 576–579.

Rose, K., Gurewitz, E., Fox, G.C., 1992. Vector quantization by deterministic annealing. IEEE Trans. Inform. Theory 38 (4), 1249–1257.

Torres, L., Huguet, J., 1994. An improvement on codebook search for vector quantization. IEEE Trans. Commun. 42 (2/3/4), 208–210.

Zeger, K., Vaisey, J., Gersho, A., 1992. Globally optimal vector quantizer design by stochastic relaxation. IEEE Trans. Signal Process. 40 (2), 310–322.