Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER
Pattern Recognition Letters 25 (2004) 1613–1617
Pattern Recognition Letters
www.elsevier.com/locate/patrec

An efficient law-of-cosine-based search for vector quantization

Kuo-Liang Chung *, Jenn-Yang Lai

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

Received 21 November 2003
Available online 19 July 2004

Abstract

Vector quantization (VQ) is a well-known compression method. In the encoding phase, given a block represented as a vector, searching the closest codeword in the codebook is a time-consuming task. An efficient law-of-cosine-based search algorithm for VQ is presented in this correspondence. In our proposed algorithm, some new formulae and a dynamic rule for selecting the fixed vector are presented to speed up the search process. Experimental results reveal that our proposed search algorithm has better execution-time when compared to the current result by Mielikainen and the previous result by Huang et al.

Keywords: Image compression; Full search; Law-of-cosines; Vector quantization

1. Introduction

Vector quantization (VQ) is an important compression technique (Gersho and Gray, 1992; Gray and Neuhoff, 1998) for low-bit-rate image compression. In the encoding phase, given a block represented as a vector, searching for the closest codeword in the codebook is a time-consuming task. For exposition, suppose we have constructed a sorted codebook with size N, say $Y = \{y_i | i = 1, 2, \ldots, N\}$ where each codeword y_i is of size k and satisfies $\|y_i\| \leqslant \|y_j\|$ for $i < j$. Given an input $\sqrt{k} \times \sqrt{k}$ block represented by a k-dimensional vector $x = (x_1, x_2, \ldots, x_k)$, the search algorithm for VQ wants to find the closest codeword y_n such that the squared Euclidean distance between the input vector x and the closest codeword $y_n = (y_{n1}, y_{n2}, \ldots, y_{nk})$ is the smallest. After finding the closest codeword in the codebook, the input vector x is replaced by the index n. Many different fast search algorithms (Kaukoranta et al., 2000; Lee and Chen, 1995; Li and Salari, 1995; Linde et al., 1980; Orchard, 1991; Ramasubramanian and Paliwal, 1992; Song and Ra, 2002) have been developed. In (Huang et al., 1992), three variants of search algorithms for VQ were presented. The first one is based on the triangle inequality. The second one is based on an inequality (see Lemma 1) to reduce the search space. The third one

* Corresponding author. Tel.: +886-227301081; fax: +886-227301080.
E-mail address: klchung@cs.ntust.edu.tw (K.-L. Chung).

0167-8655/$ - see front matter
doi:10.1016/j.patrec.2004.06.007

combines the triangle inequality and the derived inequality (see Lemma 1). Recently, according to the law-of-cosines, Mielikainen (2002) presented an efficient novel search algorithm for VQ. Using a $256 \times 256 \times 32$ AVIRIS image, experimental results reveal that Mielikainen's algorithm is superior to the GLA search algorithm (Linde et al., 1980).

An efficient law-of-cosine-based search algorithm is presented in this correspondence. The search region in VQ can be reduced significantly and it leads to the computational advantage of our proposed search algorithm. Experimental results reveal that our proposed search algorithm has better execution-time performance when compared to (Mielikainen, 2002) and (Huang et al., 1992).

2. Two past works

In this section, we survey two past works (Huang et al., 1992; Mielikainen, 2002) related to our proposed algorithm.

2.1. The work by Mielikainen

In (Mielikainen, 2002), the squared Euclidean distance between the input vector x and the codeword y_i in Y is given by

$$d_i = d(x, y_i) = \|x\|^2 + \|y_i\|^2 - 2\|x\|\|y_i\| \cos\theta_{x,i}, \quad (1)$$

where $\theta_{x,i}$ denotes the angle between the two vectors x and y_i, $-90° \leqslant \theta_{x,i} \leqslant 90°$. Arbitrarily selecting a fixed vector f, $\theta_{f,x}$ denotes the angle between f and x and $\theta_{f,i}$ denotes the angle between f and y_i, then we have $\cos\theta_{x,i} \leqslant \cos(\theta_{f,x} - \theta_{f,i})$. Let

$$d_i^* = \|x\|^2 + \|y_i\|^2 - 2\|x\|\|y_i\| \cos(\theta_{f,x} - \theta_{f,i}), \quad (2)$$

then by Eq. (1), we have $d_i^* \leqslant d_i$ because of $\cos\theta_{x,i} \leqslant \cos(\theta_{f,x} - \theta_{f,i})$.

To find the closest codeword by Eq. (2), first all the approximate distances d_i^*'s for $1 \leqslant i \leqslant N$ are calculated. If d_i^* is larger than the current minimum distance $d_{\min}$, it is unnecessary to calculate the exact distance d_i and it saves computation (Mielikainen, 2002). To reduce the time for calculating $\cos(\theta_{f,x} - \theta_{f,i})$, the following cosine identity is applied

$$\cos(\theta_{f,x} - \theta_{f,i}) = \cos\theta_{f,x}\cos\theta_{f,i} + \sin\theta_{f,x}\sin\theta_{f,i},$$

so d_i^* in Eq. (2) can be calculated by

$$d_i^* = \|x\|^2 + \|y_i\|^2 - 2\|x\|\|y_i\|\{\cos\theta_{f,x}\cos\theta_{f,i} + \sin\theta_{f,x}\sin\theta_{f,i}\},$$

where $\|y_i\|^2$, $\|y_i\|$, $\cos\theta_{f,i}$, and $\sin\theta_{f,i}$, $1 \leqslant i \leqslant N$, are calculated in advance and can be accessed using a look-up table.

2.2. The work by Huang et al.

There are three search methods presented by Huang et al. (1992). As shown in the following lemma, the inequality used in their second method can be used to eliminate some unnecessary search region in the early stage.

Lemma 1 (Huang et al., 1992). *Given an input vector x, the codeword y_i in Y is an impossible candidate to be the closest codeword of x, if the condition $(\|x\| - \|y_i\|)^2 > d_{\min}$ holds.*

3. The proposed search algorithm for VQ

In this section, we present a new pruning strategy that has two contributions: (1) we derive some new formulae to reduce the search region in order to speed up the search time and (2) instead of using the static fixed vector f for all the input vectors (Huang et al., 1992), we propose a dynamic method for selecting a more suitable fixed vector for the input vector x.

First, we find y_m from the codebook Y satisfying

$$y_m = \left\{ y_i \middle| \min_{1 \leqslant i \leqslant N} (\|x\| - \|y_i\|)^2 \right\}, \quad (3)$$

then by Eq. (3), the distance $d(x, y_m)$ is calculated as the initial minimum distance which is given by

$$d_{\min} = \|x\|^2 + \|y_m\|^2 - 2\|x\|\|y_m\| \cos\theta_{x,m}. \quad (4)$$

From Eqs. (3) and (4), the found codeword y_m is closest to the input vector x from the sense of 2-norm difference although we cannot guarantee that the condition $\|x - y_m\| = \min_{1 \leqslant i \leqslant N} \{\|x - y_i\|\}$ always holds. After all, for each new input vector

x, y_m could be selected as the current fixed vector, say f too. Since x is dynamic, we need to build up a table S for keeping these $\|y_i\|$'s in advance, $1 \leqslant i \leqslant N$, in an ascending order in order to find the fixed vector $f(=y_m)$ via the binary search method, and it leads to faster computation of $d_{\min}$. In fact, when the codebook size is N, the binary search process takes $O(\log N)$ time to find the fixed vector, and therefore saves time. For example, for a codebook with 1024 (2048) codewords, the binary search process needs at most 10 (11) steps.

Lemma 2. *Suppose the angles $\theta_{f,i}$'s for $1 \leqslant i \leqslant N$ have been calculated in advance and the angles $\theta_{f,x}$ and $\theta_{x,m}$ have also been obtained, then the codeword y_i is not a possible candidate to be the closest codeword of x if the two conditions, $|\theta_{f,x} - \theta_{f,i}| > \theta_{x,m}$ and $\|y_i\| > \|y_m\|$, hold.*

Proof. (See Appendix A)

By Lemma 2, we have the following result to reduce the search region further.

Lemma 3. *Given an input vector x, the fixed vector f is selected as y_m. The codeword y_i in Y is a possible candidate to be the closest codeword if $0 \leqslant \theta_{i,m} \leqslant 2\theta_{x,m}$ or $\|y_i\| \leqslant \|y_m\|$ holds.*

Proof. (See Appendix B)

Consequently, our proposed search algorithm consisting of four steps is listed below where the Boolean table $U[]$ is virtually used to explain the proposed algorithm, but in practice, it is unnecessary to implement the table.

Step 1. Compute $\|x\|$, $\|x\|^2$ and initialize a Boolean table $U[i] = 1$ for $1 \leqslant i \leqslant N$ to record whether the codeword y_i needs to be examined.

Step 2. Select the initial best codeword y_m from table S via the binary search method. We calculate the distance d_m denoted as $d_{\min}$. Setting the fixed vector f as y_m, we set $U[m] = 0$.

Step 3. Select the next candidate codeword y_i according to the search order (see Fig. 1; the first selected codeword to be checked is y_{m-1} and the second selected codeword is y_{m+1}), and do the following two substeps until all the values in U are 0's.

Step 3.1. By Lemma 1, if $(\|x\| - \|y_i\|)^2 \leqslant d_{\min}$, then do Step a or Step b:
- (a) While $(i > m)$, by Lemma 3, if $(\theta_{i,m} \leqslant 2\theta_{x,m})$, then calculate d_i. Replace $d_{\min}$ by d_i when $d_i < d_{\min}$. We set $U[i] = 0$.
- (b) While $(i < m)$, calculate d_i directly. Replace $d_{\min}$ by d_i when $d_i < d_{\min}$. We set $U[i] = 0$.

Step 3.2. If $(\|x\| - \|y_i\|)^2 > d_{\min}$, then do Step c or Step d:
- (c) While $(\|y_i\| \geqslant \|x\|)$, we set $U[j] = 0$ when $j \geqslant i$.
- (d) While $(\|y_i\| \leqslant \|x\|)$, we set $U[j] = 0$ when $j \leqslant i$.

Step 4. The resulting codeword y_i corresponding to $d_{\min}$ is the closest codeword of x.

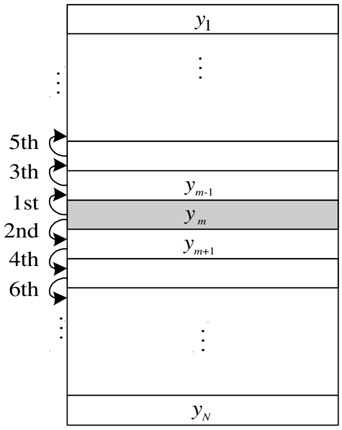

Fig. 1. The alternative search order.

Based on the above proposed algorithm, in order to speed up the search performance, we build up a table T to store all the angles $\theta_{i,j}$'s between any two different codewords y_i and y_j in Y. Under the precomputed look-up table T, after finding the suitable fixed vector for the input vector x, we can reduce the number of comparisons and arccosine calculations.

4. Experimental results

In this section, some experiments are carried out to compare the computation performance among the full search (FS), the search algorithm by Mielikainen (2002), the three search algorithms by Huang et al. (1992), and our proposed search method for VQ. The fixed vector in (Mielikainen, 2002) is selected as the mean of all the codewords in codebook. For fairness, all algorithms include the partial distortion search technique (Bei and Gray, 1985) except the FS. Two 512×512 images, Lena and Pepper, are used in the experiments. The sizes of the codebooks used are 256, 512, 1024 and 2048. The dimension of each codeword is 16. All the experiments are executed on a Pentium III PC with 700 MHz. The OS environment is Windows 2000 and the language used is Borland C++ Builder. The time unit is 10^{-3} s.

Let Mielikainen's search algorithm be abbreviated to MSA and let Huang et al.'s search algorithm be abbreviated to HSA which has three variants, namely HSA_1, HSA_2, and HSA_3, respectively. Our proposed search algorithm is denoted by OURs. Besides evaluating the true execution-time, denoted by TIME, in terms of milliseconds required by still method, for each input vector x, we still list the number of related arithmetic operations, such as the number of additions (ADD), the number of multiplications (MUL), the number of comparisons (COM), the number of square root calculations (SQR), the number of arccosine calculations (COS^{-1}), and the number of binary search process (BS). Table 1 demonstrates the computation performance among the FS, MSA, HSA, and OURs. From Table 1, when $N=256$, the execution-time improvement ratios (IR) of our proposed search algorithm over the FS, the MSA, the HSA_1, the HSA_2, and the HSA_3 are 91.9%, 43.3%, 55.2%, 48.1%, and 54.0%, respectively, where the execution-time improvement ratio is defined by

$$\mathrm{IR} = \frac{\mathrm{TIME}(A) - \mathrm{TIME}(\mathrm{OURs})}{\mathrm{TIME}(A)} \times 100\%.$$

Here the symbol A denotes the comparison algorithm. When $N=512$, the corresponding IRs are 94.0%, 45.3%, 64.5%, 58.8%, and 65.0%. Moreover, when $N=1024$, the corresponding IRs are 95.4%, 46.4%, 70.8%, 65.1%, and 71.9%. Even when $N=2048$, the corresponding IRs are 96.0%, 48.1%, 72.4%, 65.3%, and 73.5%. It comes to a conclusion that our proposed search algorithm for VQ is faster than the FS, the MSA (Mielikainen, 2002), and the three variants in (Huang et al., 1992).

Acknowledgement

This work was supported by the National Science Council of ROC under contract NSC91-2213-E011-022.

Table 1
Computation performance among FS, MSA, HSA, and OURs when $N=256$

Image	Algorithm	ADD	MUL	COM	SQR	COS^{-1}	BS	TIME	IR (%)
Lena	FS	7936	4096	255	–	–	–	2243	93.3
	MSA	1207.2	1354.3	435.3	2	–	–	308	51.2
	HSA_1	535.8	293.8	524.9	1	–	1	361	58.4
	HSA_2	682.0	488.1	464.0	1	–	1	317	52.6
	HSA_3	668.4	480.2	711.2	1	–	1	355	57.7
	OURs	297.5	168.0	152.3	1	1	1	150	–
Pepper	FS	7936	4096	255	–	–	–	2263	90.6
	MSA	1260.1	1367.7	458.5	2	–	–	329	35.5
	HSA_1	729.7	393.4	624.5	1	–	1	443	52.1
	HSA_2	840.5	568.9	544.9	1	–	1	376	43.6
	HSA_3	826.7	561.2	792.2	1	–	1	428	50.4
	OURs	465.3	255.6	247.7	1	1	1	212	–

Appendix A. The proof of lemma 2

Proof. From Eq. (3), we know $y_m = \{y_i | \min_{1 \leqslant i \leqslant N} (\|x\| - \|y_i\|)^2\}$. For all the other codewords y_i's in Y, we have

$$(\|x\| - \|y_i\|)^2 \geqslant (\|x\| - \|y_m\|)^2.$$

Equivalently, it yields

$$\begin{aligned}(\|x\| - \|y_i\|)^2 &- (\|x\| - \|y_m\|)^2 \\ &= \|y_i\|^2 - \|y_m\|^2 + 2\|x\|(\|y_m\| - \|y_i\|) \geqslant 0.\end{aligned} \tag{A.1}$$

By Eq. (1) and the law-of-cosines, we have

$$\begin{aligned} d_m &= \|x\|^2 + \|y_m\|^2 - 2\|x\|\|y_m\|\cos\theta_{x,m} \\ &\geqslant \|x\|^2 + \|y_m\|^2 - 2\|x\|\|y_m\|\cos(\theta_{f,x} - \theta_{f,m}) \end{aligned}$$

and

$$\begin{aligned} d_i &= \|x\|^2 + \|y_i\|^2 - 2\|x\|\|y_i\|\cos\theta_{x,i} \\ &\geqslant \|x\|^2 + \|y_i\|^2 - 2\|x\|\|y_i\|\cos(\theta_{f,x} - \theta_{f,i}), \end{aligned}$$

where $1 \leqslant i \leqslant N$, then it is unnecessary to calculate d_i if the condition

$$\begin{aligned} &\|x\|^2 + \|y_i\|^2 - 2\|x\|\|y_i\|\cos(\theta_{f,x} - \theta_{f,i}) \\ &\quad > \|x\|^2 + \|y_m\|^2 - 2\|x\|\|y_m\|\cos\theta_{x,m}, \end{aligned}$$

i.e.,

$$\begin{aligned} &\|y_i\|^2 - \|y_m\|^2 + 2\|x\|\|y_m\|\cos\theta_{x,m} \\ &\quad - 2\|x\|\|y_i\|\cos(\theta_{f,x} - \theta_{f,i}) > 0, \end{aligned} \tag{A.2}$$

holds. Considering the case when the left side of Eq. (A.2) is larger than the left side of Eq. (A.1), if we subtract the left side of Eq. (A.1) from the left side of Eq. (A.2), we have

$$\begin{aligned} &2\|x\|\{\|y_m\|\cos\theta_{x,m} - \|y_i\|\cos(\theta_{f,x} - \theta_{f,i})\} \\ &\quad - 2\|x\|\{\|y_m\| - \|y_i\|\} > 0. \end{aligned}$$

After some algebraic simplifications, we have

$$\|y_i\|\{1 - \cos(\theta_{f,x} - \theta_{f,i})\} - \|y_m\|\{1 - \cos\theta_{x,m}\} > 0.$$

On the other hand, it is unnecessary to calculate d_i if the two conditions, $|\theta_{f,x} - \theta_{f,i}| > \theta_{x,m}$ and $\|y_i\| > \|y_m\|$, hold. We complete the proof. □

Appendix B. The proof of lemma 3

Proof. From the opposite meaning of Lemma 2, the codeword y_i is a possible candidate to be the closest codeword if $|\theta_{f,x} - \theta_{f,i}| \leqslant \theta_{x,m}$, i.e., $\theta_{f,x} - \theta_{x,m} \leqslant \theta_{f,i} \leqslant \theta_{f,x} + \theta_{x,m}$, or $\|y_i\| \leqslant \|y_m\|$ holds. Since f is selected as y_m, then $\theta_{f,x}$ is equal to $\theta_{x,m}$. The equation $\theta_{f,x} - \theta_{x,m} \leqslant \theta_{f,i} \leqslant \theta_{f,x} + \theta_{x,m}$ can be simplified to $0 \leqslant \theta_{i,m} \leqslant 2\theta_{x,m}$. We complete the proof. □

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.

Gersho, A., Gray, R.M., 1992. Vector Quantization and Signal Compression. Kluwer Academic Pub., Boston.

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

Huang, C.M., Bi, Q., Stiles, G.S., Harris, R.W., 1992. Fast full search equivalent encoding algorithms for image compression using vector quantization. IEEE Trans. Image Process. 1 (3), 413–416.

Kaukoranta, T., Franti, P., Nevalainen, O., 2000. A fast exact GLA based on code vector activity detection. IEEE Trans. Image Process. 9 (8), 1337–1342.

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.

Li, W., Salari, E., 1995. A fast vector quantization encoding method for image compression. IEEE Trans. Circ. Systems Video Technol. 5 (2), 119–123.

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

Mielikainen, J., 2002. A novel full-search vector quantization algorithm based on the law of cosines. IEEE Signal Process. Lett. 9 (6), 175–176.

Orchard, M.T., 1991. A fast nearest-neighbor search algorithm. IEEE Internat. Conf. Acoust., Speech, Signal Process. 4, 2297–2300.

Ramasubramanian, V., Paliwal, K.K., 1992. Fast k-dimensional tree algorithms for nearest neighbor search with application to vector quantization encoding. IEEE Trans. Signal Process. 40 (3), 518–531.

Song, B.C., Ra, J.B., 2002. A fast search algorithm for vector quantization using L_2-norm pyramids of codewords. IEEE Trans. Image Process. 11 (1), 10–15.