Transactions Briefs

Extended Mean-Distance-Ordered Search Using Multiple ℓ_1 and ℓ_2 Inequalities for Fast Vector Quantization

Sun-Young Choi and Soo-Ik Chae

***Abstract*—Mean-distance-ordered search (MOS) algorithms [1], [2] were proposed as a fast and efficient method for vector quantization (VQ) encoding. However, the efficiency of MOS algorithms is limited because they use only one inequality in reducing the search region. Therefore, we propose an extended MOS algorithm using multiple anchor vectors which exploits multiple ℓ_1 and ℓ_2 inequalities to reduce the search region further. We also describe a greedy algorithm for selecting an anchor vector set that reduces the computational cost of the extended MOS algorithm. For eight test images, the number of operations required in the extended MOS algorithm was 66.2% of that in the previous MOS algorithm on the average when the codebook size is 256, while producing the same encoding quality to that of the full-search VQ.**

***Index Terms*—Anchor vector, mean-distance-ordered search, norm, triangular inequality, vector mean, vector quantization..**

I. Introduction

Vector quantization (VQ) is an efficient data-encoding technique that exploits dependencies among vector components [9]. The full-search VQ finds the nearest code vector that has the minimum distortion from an input vector by exhaustively searching all code vectors in the codebook. Because its computational cost is so high, its utilization is limited. Therefore, many fast VQ algorithms have been developed that reduce the computational cost substantially while producing the same reconstructed image quality as the full-search VQ.

These fast algorithms can be grouped into three categories: partial distortion elimination (PDE) [3], triangular inequality elimination (TIE) [4]–[6], and mean-distance-ordered search (MOS) [1], [2]. A TIE-based VQ using multiple anchor vectors that were called control vectors in [5] can further reduce the search region by intersecting the search regions for all anchor vectors [9]. An anchor vector is a reference point from which its distance to each code vector is precomputed and stored. After eliminating some of the code vectors by using the inequality for an anchor vector, we can obtain a subset of the codebook that contains the nearest code vector to the given input vector, which is called the search region for the anchor vector. However, the mean-distance-ordered search (MOS)-based VQ [1], [2] exploits only one inequality between the vector mean-distance and the distortion of input and code vectors.

Therefore, the aim of this paper is to propose an extended MOS algorithm that exploits the ℓ_1 and ℓ_2 inequalities for multiple anchor vectors. In Section II, the conventional MOS algorithm is described briefly. In Section III, we first derive a generalized inequality between the distortion and the vector mean-distance relative to an anchor vector by using an ℓ_1 and ℓ_2 inequality. Because an inequality can be obtained for each anchor vector, we introduce an extended MOS algorithm using multiple inequalities for an anchor vector set. Although a VQ based on the extended MOS algorithm can further reduce the search region for each input vector, the reduction in its computational cost depends on the anchor vectors. Therefore, we also propose a greedy algorithm for selecting an anchor vector set that minimizes the computational cost of the extended MOS algorithm in Section IV. Simulation results are presented in Section V, followed by the conclusion in Section VI.

Manuscript received August 1998; revised December 1999. This paper was recommended by Associate Editor J. Bioucas-Dias.

S.-Y. Choi was with the Institute of Advanced Engineering, Seoul National University, Kwanak-Gu, Seoul, Korea. He is now with the Institute for Advanced Engineering, Yongin, Kyongi-Do 449–860, Korea.

S.-I. Chae is with the School of Electrical Engineering, Seoul National University, Kwanak-Gu, Seoul, Korea.

Publisher Item Identifier S 1057-7130(00)03131-1.

II. MOS

For an input vector $x = (x_1, x_2, \cdots, x_n)$ and a code vector $c_i = (c_{i1}, c_{i2}, \cdots, c_{in})$, the MOS algorithm [1], [2] uses the inequality between the mean distance and the distortion

$$\left(\sum_{k=1}^{n} x_k - \sum_{k=1}^{n} c_{ik}\right)^2 \le n \cdot d^2(x, c_i) \tag{1}$$

where n is the dimension of the input and code vectors and $d(x, c_i) = (\sum_{k=1}^{n} (x_k - c_{ik})^2)^{1/2}$. Hereafter, we will call $\sum_{k=1}^{n} x_k$ the mean of a vector x although it is the sum of the vector components. In addition, we also refer to the left-hand side of (1) as the squared mean-distance (SMD) between x and c_i just to follow the convention in [2].

In the MOS algorithms, the code vectors are pre-ordered according to their vector means. For a given input vector x, the search procedure finds the nearest code vector in the ℓ_2 distance sense, starting from the initial code vector, which is the nearest code vector to x in the squared mean-distance sense. Then, the procedure moves up and down from the initial code vector alternately in the preordered list. Let c_i be the code vector having the current minimum distortion, $d(x, c_i)$, among the code vectors searched so far. If a code vector c_j satisfies the inequality

$$n \cdot d^2(x, c_i) \le \left(\sum_{k=1}^{n} x_k - \sum_{k=1}^{n} c_{jk}\right)^2 \tag{2}$$

we do not need to calculate $d(x, c_j)$ because $d(x, c_j) \ge d(x, c_i)$. Furthermore, we do not check (2) for all the code vectors, c_k that are further away from c_i than c_j in the pre-ordered list because their SMD's are larger than $n \cdot d^2(x, c_i)$.

III. Extension of Inequality Using Relative Vector Means

First, the following inequality between $\ell_1(x)$ and $\ell_2(x)$ is always satisfied [7]:

$$\ell_1(x) \le \sqrt{n} \cdot \ell_2(x) \tag{3}$$

where a p-norm of an n-dimensional vector x denoted as $\ell_p(x)$, is defined by

$$\ell_p(x) = \|x\|_p = \left(|x_1|^p + \cdots + |x_n|^p\right)^{1/p}, \qquad p \ge 1. \tag{4}$$

Assume that a set A of anchor vectors is given, which was introduced in the TIE-based VQ [5], [6], [9]. For each anchor vector $a_k = (a_{k1}, a_{k2}, \cdots, a_{kn})$ in A, we define a relative vector mean of x to an anchor vector a_k as $\ell_1(x - a_k)$. The relative vector mean is useful in

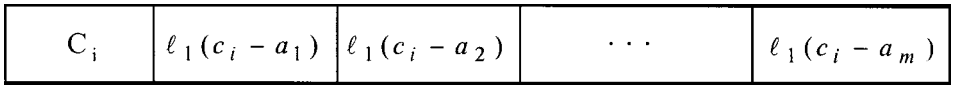

Fig. 1. Relative vector means to m anchor vectors $\{a_1, a_2, \cdots, a_m\}$ are precalculated and stored for each code vector c_i.

deriving an inequality for each anchor vector. Using (3) and the triangular inequality for ℓ_1, we obtain

$$(\ell_1(x - a_k) - \ell_1(c_i - a_k))^2 \le n \cdot d^2(x, c_i). \quad (5)$$

Inequality (5) is a simple extension of (1) for an arbitrary anchor vector, which reduces to (1) when $a_k = (0, 0, \cdots, 0)$.

For an input vector, x, and its initial code vector, c_i, we can obtain a search region $S(a_k, c_i, x) = \{c_j | (\ell_1(x - a_k) - \ell_1(c_j - a_k))^2 < n \cdot d(x, c_i)^2$ for $c_j \in \Omega\}$ for each anchor vector, a_k in A. Here, Ω is the codebook. Therefore, we can obtain the reduced search region S, which is the intersection of the search regions $S(a_k, c_i, x)$, for all $a_k \in A$.

We define the principal anchor vector as an anchor vector whose inequality is tested first. In the extended MOS algorithm, the code vectors are preordered in ascending order of their relative vector means for the principal anchor vector a_1. Furthermore, for each code vector, its relative means to all other anchor vectors are also pre-calculated and stored, as shown in Fig. 1.

For a codebook Ω and an anchor vector set A, the extended MOS algorithm searches the pre-ordered codebook, as described in the following pseudo code, where m and N are the number of the anchor vectors and the codebook size, respectively.

```
Get an input vector x;
Find its initial code vector c_i with a binary search;
j = i; c_min = c_i; Up = Down = true;
while (((Up == true) or (Down == true))
and ((i > 1) or (j < N))) {
  if ((Up == true) and (j < N)) then
{j = j + 1; /* index for the upper search */
    if ((ℓ1(x − a1) − ℓ1(cj − a1))^2 ≥ n · d^2(x, c_min))
then Up = false;
    else { for (k = 2; k ≤ m and (ℓ1(x − ak) −
ℓ1(cj − ak))^2 < n · d^2(x, c_min); k = k + 1);
    if ((k == m + 1) and (d(x, cj) < d(x, c_min)))
then c_min = cj; } }
if ((Down == true) and (i > 1)) then {i = i −
1; /* index for the lower search */
    if ((ℓ1(x − a1) − ℓ1(ci − a1))^2 ≥ n · d^2(x, c_min))
then Down = false;
    else { for (k = 2; k ≤ m and (ℓ1(x − ak) −
ℓ1(ci − ak))^2 < n · d^2(x, c_min); k = k + 1);
    if ((k == m + 1) and (d(x, ci) < d(x, c_min)))
then c_min = ci; } }
} /* end of while */.
```

IV. Selecting the Anchor Vector Set

We now describe a greedy algorithm for selecting the anchor vector set that minimizes the computational cost, which greatly affects the performance of the extended MOS algorithm. To reduce the computational complexity of the greedy algorithm, we use the codebook Ω as a good representative of the input vector space because the codebook is much smaller than the input vector space L^n, where $L = \{0, 1, \cdots, 255\}$. By the same token, we also limit the candidate anchor vector set C to $\Omega \cup V$, from which we select the anchor vectors. Here, V is the subset of all the vertices in L^n, which is

$$V = \{(b_1, b_1, b_2, b_2, \cdots, b_{n/2}, b_{n/2}) \cdot | b_i \in \{0, 255\}, 1 \le i \le n/2\}. \quad (6)$$

The reason of including V in C is that the vertices are good candidates of anchor vectors, which was confirmed with the experimental results. We found that the principal anchor vector is usually the zero vector (0, 0, $\cdots$, 0) in the simulations we had done.

For a given input vector, x, first we find its initial code vector c_i in the preordered list of the code vectors for the principal anchor vector a_1. Then, we do not need to calculate the distance $d(x, c_j)$ for any code vector c_j that satisfies the inequality $n \cdot d^2(x, c_i) \le (\ell_1(x - a_1) - \ell_1(c_j - a_1))^2$ Here, the SMD of x and c_j for the principal anchor vector a_1 is a lower bound of $n \cdot d^2(x, c_j)$. When the SMD of x and c_j is a tighter lower bound of $n \cdot d^2(x, c_j)$, it is more probable to reject the code c_j that is not nearer to x than c_i in the ℓ_2 distance sense so that we need not calculate its distance to x.

Therefore, we select a vector $a \in C$ as an anchor vector, such that the vector a minimizes $n \cdot d^2(x_p, c_j) - (\ell_1(x_p - a) - \ell_1(c_j - a))^2$ on the average for $c_j, x_p \in \Omega$. The cost function $f_1(\cdot)$ for selecting the principal anchor vector is expressed as

$$f_1(a) = \sum_{p=1}^{N} \sum_{j=1}^{N} \left[n \cdot d^2(x_p, c_j) - (\ell_1(x_p - a) - \ell_1(c_j - a))^2 \right]. \quad (7)$$

Then, the principal anchor vector is determined as

$$a_1 = \arg\min_{a \in C} (f_1(a)).$$

We used another measure in selecting the other anchor vectors. First, for an anchor vector a_k and a vector $x_p \in \Omega$, we define the minimal search region $S_{\min}(a_k, x_p)$ as a search region obtained when the nearest code vector c_l to x_p is used as its initial code vector

$$S_{\min}(a_k, x_p) = \{c_j | (\ell_1(x_p - a_k) - \ell_1(c_j - a_k))^2 < n \cdot d^2(x_p, c_l), c_j \in \Omega\} \quad (8)$$

where

$$c_l = \arg\min_{c_k \in \Omega, c_k \ne x_p} \{d(c_k, x_p)\}.$$

Assuming that a set of pre-selected $(k-1)$ anchor vectors $A_{k-1} = \{a_1, a_2, \cdots, a_{k-1}\}$ is given, we select another vector $a \in C - A_{k-1}$ as the next kth anchor vector such that the vector a minimizes the cardinality sum of the intersection of the k minimal regions over all $x_p \in \Omega$, which is a greedy algorithm. The cost function $f_2(\cdot)$ for selecting the kth anchor vector is represented as

$$f_2(k, a) = \frac{1}{N} \sum_{p=1}^{N} \left| \left(\bigcap_{i=1}^{k-1} S_{\min}(a_i, x_p) \right) \cap S_{\min}(a, x_p) \right| \quad (9)$$

TABLE I
COMPUTATIONAL COSTS FOR THREE TEST IMAGES WHEN THE CODEBOOK SIZE IS 256

Image	Fast VQ	Multiply	Add/sub	Compare	Total op.
All	Full-search	256.0	496.0	16.0	768.0
Jaguar	Li's TIE (3)	20.2	39.1	11.1	70.4
	Ra's MOS	26.2	50.1	3.6	79.9
	Extended MOS (3)	14.9	30.9	5.1	50.8
Pepper	Li's TIE (3)	12.3	23.9	7.4	43.6
	Ra's MOS	16.8	32.5	2.5	51.8
	Extended MOS (3)	8.5	19.9	3.3	31.6
Lena	Li's TIE (3)	13.0	25.0	8.0	46.0
	Ra's MOS	17.3	33.4	2.5	53.1
	Extended MOS (3)	8.8	20.4	3.4	32.6

where $|C|$ is the cardinality of a set C. Then, the kth anchor vector is determined as

$$a_k = \arg\min_{a \in C - A_{k-1}} (f_2(k, a)), \quad \text{for all } k = 2, \cdots, m. \tag{10}$$

In the extended MOS algorithm with m anchor vectors, the average number of multiplications per a vector component (a pixel) required in encoding an input vector can be roughly estimated with

$$g(m) \approx f_2(m, a_m) + \frac{1}{n} \cdot \sum_{k=1}^{m} f_2(k, a_k) \tag{11}$$

where n and m are the dimension of the vector and the number of selected anchor vectors, respectively. The first term is the expectation of the cardinality of the intersection of m minimal search regions for the representative input vector space. The second term is the upper bound of the expectation of the SMD computation divided by the dimension of the vector because we stop checking the inequalities $(\ell_1(x - a_k) - \ell_1(c_j - a_k))^2 < n \cdot d^2(x, c_{\min})$ for $a_k \in A$ if the inequality for any one of the anchor vectors is not satisfied. From (11), we can estimate the optimal number of anchor vectors as

$$m = \arg\min_{m}(g(m)).$$

V. SIMULATION RESULTS

A codebook Ω was generated with the Linde–Buzo–Gray algorithm [8] using training images of 512×512 size such as Boats, Bridge, Sail, Crowd, and Einstein. Codebook size is 256 and vector dimension is 16. As explained in Section IV, the anchor vectors were selected by using (7) and (9). The computational cost of the extended MOS algorithm was compared to those of the full-search (FS) VQ, Ra's MOS algorithm [2] and Li's TIE-based VQ [5]. Here, the number of operations per pixel was used to measure the computational cost in all the simulation results.

As shown in Table I, the computational cost of the extended MOS algorithm is reduced substantially for all the test images, compared to the other algorithms. The digit in the parenthesis indicates the number of anchor vectors used. Note that three anchor vectors were selected in the extended MOS VQ as well as in the Li's TIE-based VQ. The former is better than the latter, because the number of compare operations is reduced to about half and the calculation of the $\ell_1 - \ell_2$ distance is much simpler than that of the ℓ_2 distance.

The computational costs for the fast VQ algorithms, which were averaged for eight test images such as Lena, F16, Jaguar, Pepper, Couple, Girl, Baboon, and Zelda, are shown in Table II. In the extended MOS algorithm, the number of operations of multiply, and add/subtract operations were reduced by 42.7% and 35.1%, respectively, while that of compare operations is increased by 40.0%, when compared to Ra's MOS algorithm [2]. However, note that the portion of compare operation is small in the total operations. Assume that all types of operations have equal computational cost although multiplication is more complex than addition or comparison. Then, the total number of operations in the extended MOS algorithm is reduced to 66.2%, when compared to Ra's MOS algorithm.

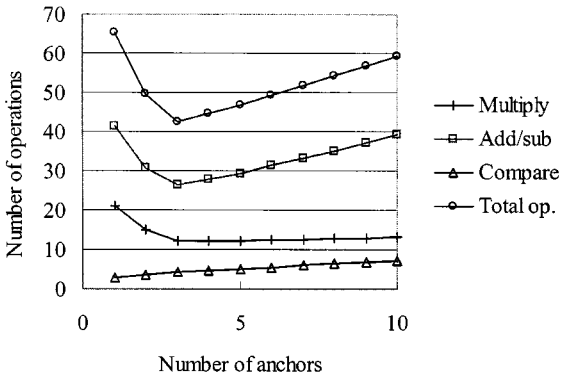

Fig. 2. Average number of operations versus the number of anchor vectors in the extended MOS algorithm when the codebook size is 256.

TABLE II
AVERAGE COMPUTATIONAL COSTS FOR EIGHT TEST IMAGES WHEN THE CODEBOOK SIZE IS 256

	Multiply	Add/sub	Compare	Total op.
Full-search	256.0	496.0	16.0	768.0
Li's TIE (3)	16.1	31.1	9.1	56.4
Ra's MOS	21.1	40.5	3.0	64.5
Extended MOS (3)	12.1	26.3	4.2	42.7

(a) (b) (c)

Fig. 3. (a) Original Lena image (512 × 512). (b) Reconstructed image of full-search VQ (PSNR : 31.75 dB). (c) Reconstructed image of extended MOS VQ (PSNR : 31.75 dB), when the codebook size is 1024.

If more anchor vectors are used, the cardinality of the search region is reduced further although the number of the SMD computation is increased, as shown in Fig. 2. From the simulation results, we found that three anchor vectors was optimal for the codebook of 256 codes with dimension 16. We also compared the computational cost for fast VQ algorithms when the codebook size was changed from 256 to 1024, as shown in Table III. In Table III, the computational complexity of the extended MOS using three anchor vectors is reduced substantially, compared to that using only the principal anchor vector. We found that the percentage of reduction in the number of multiplication in the extended MOS algorithm was increased from 43% to 51% as the codebook size increases from 256 to 1024, compared to Ra's MOS algorithm [2].

Fig. 3 shows the original Lena image and the reconstructed images of full-search VQ and the extended MOS VQ when codebook size is 1024. The extended MOS produces the same VQ coding quality to that

TABLE III
AVERAGE COMPUTATIONAL COSTS FOR THE CODEBOOK SIZES FROM 256–1024

	Multiply			Total op.		
Codebook size	256	512	1024	256	512	1024
Full-search	256.0	512.0	1024.0	768.0	1536.0	3072.0
Li's TIE (3)	16.1	27.3	48.0	56.4	96.8	172.8
Ra's MOS	21.1	38.4	71.8	64.5	116.7	216.9
Extended MOS (1)	21.1	38.4	71.8	64.5	117.6	216.9
Extended MOS (3)	12.1	20.5	35.1	42.7	67.9	111.7

of the full-search VQ but reduces the total operations to about 3.6% at the codebook size of 1024.

In the conventional MOS algorithm [2], the memory of $bit\_width \times N$ bits for storing $\ell_1(c_i)$ for all i is required in addition to the memory for the codebook. In contrast, the additional memory for storing $\ell_1(c_i - a_k)$ for all i and k is $bit\_width \times m \times N$ bits in the extended MOS algorithm, which is proportional to the number of anchor vectors. We found that if bit_width was less than 12, the image quality was degraded.

VI. CONCLUSION

By introducing a new concept of the relative vector mean, we extended the MOS algorithm [1], [2] so that the inequalities for multiple anchor vectors can be exploited. In the extended MOS algorithm, we reduced the computational cost substantially by reducing the search region further with multiple inequalities. The extended MOS algorithm required only 66.2% of the number of operations used in the previous MOS algorithm, while producing the same encoding quality as the full-search VQ. We also proposed a greedy algorithm for selecting the anchor vectors by using the cost functions based on the cardinality of the minimal search region. Furthermore, we also explained how to determine the number of the anchor vectors with a formula that estimates the number of multiply operations. The contribution of this brief is to show the extensibility of the MOS in fast VQ by introducing the anchor vectors.

REFERENCES

[1] G. Poggi, "Fast algorithm for full-search VQ encoding," *Electron. Lett.*, vol. 29, pp. 1141–1142, June 1993.

[2] S.-W. Ra and J.-K. Kim, "A fast mean-distance-ordered partial codebook search algorithm for image vector quantization," *IEEE Trans. Circuits Syst. II*, vol. 40, pp. 576–579, Sept. 1993.

[3] C. D. Bei and R. M. Gray, "An improvement of the minimum distortion encoding algorithm for vector quantization," *IEEE Trans. Commun.*, vol. COM-33, pp. 1132–1133, Oct. 1985.

[4] 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, pp. 413–416, July 1992.

[5] W. Li and E. Salari, "A fast vector quantization encoding method for image compression," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, pp. 119–123, Apr. 1995.

[6] S.-Y. Choi and S.-I. Chae, "Incremental-search fast vector quantizer using triangular inequalities for multiple anchors," *Electron. Lett.*, vol. 34, no. 12, pp. 1192–1193, June 1998.

[7] G. H. Golub and C. F. Loan, *Matrix Computations*, 3rd ed. Baltimore, MD: The Johns Hopkins Press, 1996.

[8] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. COM-28, pp. 84–95, Jan. 1980.

[9] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Norwell, MA: Kluwer, 1992.

MVDR Beamforming and Generalized Sidelobe Cancellation Based on Inverse Updating with Residual Extraction

Marc Moonen and Ian K. Proudler

***Abstract*—The "standard" minimum variance distortionless response (MVDR) beamforming algorithm of McWhirter and Shepherd [5] is known to suffer from linear roundoff error buildup. An alternative [7] has been developed based on the so-called inverse QR-updating algorithm. It is amenable to pipelined implementation, but unlike the McWhirter and Shepherd algorithm, it is stable numerically. Here, a variant of the latter algorithm that combines (part of) the McWhirter and Shepherd procedure into the inverse QR updating scheme is discussed. This makes the derivation of the algorithm less complex compared to the original [7]. Furthermore, by making use of a residual extraction property of the inverse updating procedure, an interesting correspondence of the new MVDR algorithm with so-called 'generalized sidelobe cancellation' is explicitly revealed.**

I. INTRODUCTION

The minimum variance distortionless response (MVDR) beamforming problem amounts to minimising, in a least-squares sense, the combined output from an antenna array subject to K independent linear equality constraints, each of which corresponds to a given 'look direction'. By "independent," we mean that the minimum array output is computed for each constraint in turn. In other words, K independent recursive least-squares problems have to be solved at once. The aim is to derive efficient (parallel) algorithms for this.

In [4], a parallel solution is given for the linearly constrained recursive least-squares problem, with a constraint *pre*-processor coupled to a Gentleman-Kung triangular array [1]. For MVDR beamforming, one would then need K such triangular arrays, which is inefficient. The so-called "generalized sidelobe canceller" of [2] is also based on constraint pre-processing, and hence, equally unsuitable for the MVDR problem. In [5], however, McWhirter and Shepherd have shown how the beamforming problem can be solved with only one triangular array, coupled to a constraint *post*-processor. This is commonly accepted as

Manuscript received November 1998; revised December 1999. This work supported by the Belgian State, Prime Minister's Office (F.D.W.T.C), Interuniversity Poles of Attraction Programme IUAP P4-02, the Concerted Research Action MIPS of the Flemish Government, Research Project FWO nr. G.0295.97. The work of the second author was supported by the Technology Group TG10 of the MoD Corporate Research Programme.This paper was recommended by Associate Editor J. Bioucas-Dias

M Moonen is with K.U. Leuven, Electrical Engineering Department, 3001 Heverlee, Belgium.

I. K. Proudler is with DERA, St Andrews Road, Malvern, WR14 3PS, U.K.

Publisher Item Identifier S 1057-7130(00)03120-7.