A FAST SEARCH METHOD FOR VECTOR QUANTIZATION USING 2-PIXEL-MERGING SUM PYRAMID IN RECURSIVE WAY

Zhibin Pan[1)], Koji Kotani[2)], and Tadahiro Ohmi[1)]

1) New Industry Creation Hatchery Center, Tohoku University, Japan
2) Department of Electronic Engineering, Graduate School of Engineering, Tohoku University, Japan
Aza-aoba 05, Aramaki, Aoba-ku, Sendai, 980-8579, Japan
E-mail: pzb@fff.niche.tohoku.ac.jp

ABSTRACT

Vector quantization (VQ) is a classical but still very promising signal compression method. In the framework of VQ, fast search method is a key issue because it is the time bottleneck in VQ encoding process. To speed up VQ, some fast search methods that are based on a 4-pixel-merging (4-PM) mean pyramid data structure have already been proposed in previous works [5], [6]. However, during the search process, the methods in these previous works discard the obtained value of Euclidean distance at an intermediate level completely if a rejection test fails at this level. This discarded value is a waste to the computation and becomes the overhead, which will certainly degrade the overall search efficiency of VQ encoding.

To solve the overhead problem of computation, this paper proposes a 2-pixel-merging sum pyramid data structure and a recursive way for computing Euclidean distances level by level, which can reuse the obtained value of Euclidean distance at any level thoroughly to compute the next rejection test condition at a successive level. Mathematically, the proposed method can overcome the overhead problem of computation completely and reduce the computational burden that is needed in a conventional non-recursive way to about half at each level. Experimental results confirmed the proposed method outperforms the previous works obviously.

1. INTRODUCTION

Vector quantization (VQ) is a widely used asymmetric signal compression method [1]. In conventional VQ method, an N×N image to be encoded is firstly divided into a series of non-overlapping smaller n×n image blocks. Then VQ encoding is implemented block by block sequentially. The distortion between an input image block and a codeword can be measured by squared Euclidean distance for simplicity as

$$d^2(I, C_i) = \sum_{j=1}^{k} \left(I_j - C_{i,j}\right)^2 \qquad i = 1,2,\cdots,N_c \qquad (1)$$

where I is the current image block, C_i is the i^{th} codeword, j represents the j^{th} element of a vector, k (=n×n) is the vector dimension and N_c is the codebook size.

Then a best-matched codeword with minimum distortion, which is called winner afterwards, can be determined straightforwardly by

$$d^2(I, C_w) = \min_i \left[d^2(I, C_i)\right] \qquad i = 1,2,\cdots,N_c \qquad (2)$$

where C_w means winner. And subscript "w" is the index of the winner. This process for finding winner is called full search (FS) because the matching is executed over the whole codebook. FS is very heavy computationally due to it performing N_c times k-dimensional Euclidean distance computation. Once "w" has been found, which conventionally uses much less bits than an original codeword, VQ only transmits this index "w" instead of the codeword C_w to reduce the amount of image data in order to realize image compression. Because the same codebook has also been stored at the receiver, by using the received index "w", it is very easy to reconstruct an image by pasting the corresponding codword one by one.

Based on the mechanism of VQ described above, it is clear that VQ has a very heavy encoding process and a very simple decoding process. The fast search method in encoding process limits the performance of VQ to a great extent. In practice, VQ encoding method is especially applicable to a broadcasting-type low-cost communications system.

Because a high dimension k is the main problem for fast VQ encoding, it is very important to use low dimensional features to approximately express a vector in order to reject obviously unlikely codewords. There exist many fast search methods for VQ. One class of them [2]-[4] is based on using scalar statistical features of a vector (L_1 norm, the variance and L_2 norm), which can roughly describe a vector with low dimension, to compute the difference between two vectors in order to reject a candidate codeword to avoid computing Euclidean distance. Another class of them [5], [6] is based on the multi-resolution concept by constructing an appropriate pyramid data structure to roughly describe the original vector with a series of low dimensional levels in order to make a rejection. Both classes of methods are very search-efficient but they suffer from the overhead problem of computation once a rejection test fails. This paper aims at improving the method proposed in [5], [6] and overcoming the overhead problem of computation completely.

2. RELATED PREVIOUS WORKS

During a winner search process, suppose the "so far" minimum Euclidean distance is d_{min}. Based on statistical features of a vector, the previous work [2] proposed a codeword rejection rule as: If $k(MI - MC_i)^2 \geq d_{\min}^2$ holds, then reject C_i safely, where MI is the mean of an input image block I and MC_i means the same for C_i. This is the famous ENNS method. Then, the previous work [3] proposed a supplementary codeword rejection rule when ENNS method fails as: If $k(MI - MC_i)^2 + (VI - VC_i)^2 \geq d_{\min}^2$ holds, then reject C_i safely as well, where VI is the variance of I and VC_i means the same for

C_i. This is the famous EENNS method. To improve [3] further, the previous work [4] proposed another supplementary codeword rejection rule when EENNS method also fails as: If $(L_2 I - L_2 C_i)^2 \geq d^2_{\min}$ holds, then reject C_i safely as well, where $L_2 I$ is the L_2 norm of I and $L_2 C_i$ means the same for C_i. This is state-of-the-art EEENNS (i.e. equal-average equal-variance equal-norm nearest neighbor search) method. EEENNS method needs three extra memories for each codeword. Obviously, if any one of these three rejection tests fails, it will inevitably become the overhead of computation.

On the other hand, a 4-pixel-merging (4-PM) mean pyramid data structure as shown in Fig.1 (a) has been proposed in [5], [6] to realize a multi-resolution description for a vector.

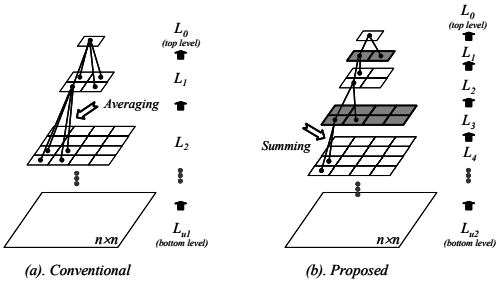

Fig.1. For an n×n block, (a) a 4-PM mean pyramid with bottom level u1=$\log_4$(n×n) and (b) a 2-PM sum pyramid with bottom level u2=$\log_2$(n×n)=2×u1. One new level is sandwiched in-between every 2 levels of a conventional 4-PM pyramid (shadowed).

Then a hierarchical rejection rule [5], [6] is set up as

$$d^2_{m,u1}(I,C_i) \geq \cdots \geq 4^{u1-v} d^2_{m,v}(I,C_i) \geq \cdots \geq 4^{u1-1} d^2_{m,1}(I,C_i) \geq 4^{u1} d^2_{m,0}(I,C_i) \quad (3)$$

Suppose $MI_{v,m}$ is the m^{th} pixel (the mean after roundoff) at the v^{th} level in a 4-PM mean pyramid for I and $MC_{i,v,m}$ implies the same thing to C_i for m∈[1~4^v], then Euclidean distance at the v^{th} level for v∈[0~u1] between the two 4-PM mean pyramids is $d^2_{m,v}(I,C_i) = \sum_{m=1}^{4^v} (MI_{v,m} - MC_{i,v,m})^2$, where the real Euclidean distance $d(I, C_i)=d_{m,u1}(I, C_i)$ holds (based on the definition in Eq.1) and u1=2 for a 4×4 block.

At any v^{th} level for v∈[0~u1], if $4^{u1-v} d^2_{m,v}(I, C_i) > d^2_{\min}$ holds, then the current real Euclidean distance $d^2_{m,u1}(I, C_i)$ would be definitely larger than $d^2_{\min}$. Thus, the search can be terminated at this v^{th} level and C_i can be rejected safely.

Originally, this 4-PM mean pyramid data structure is used for a progressive image transmission. In this (u1+1) level mean pyramid, each pixel value is computed by averaging the neighboring 2×2 pixels at the corresponding lower level and then taking the roundoff. Therefore an image block can be viewed with (u1+1) resolutions and then transmitted progressively from the top towards the bottom level by level. If the transmission can be terminated before it reaches the bottom L_{u1} level by a practical requirement at the receiver, image compression can be realized because the transmitted maximum data amount so far will be $(1+4^1+4^2+ \cdots +4^{u1-1})= (4^{u1}-1)/3=(k-1)/3$. However, all of the data above will become the overhead if the bottom L_{u1} level is transmitted. Obviously, there are two constraints for constructing a pyramid for progressive image transmission. First, it is necessary to guarantee that each pixel value is an integer and in [0, 255] interval in order to display it by 8-bit encoding. Second, it is also necessary to keep the aspect ratio of a 2-dimensional image to be constant in order to show it correctly. Therefore, the averaging, the roundoff operation and 2×2 pixel merging way are required.

The 4-PM mean pyramid method needs $(1+4^1+4^2+ \cdots +4^{u1-1})$ extra memories for each codeword. This is much more than EEENNS method. Obviously, if a rejection test fails at a level, the overhead of computation will inevitably occurs as well. This is because when a test fails at the v^{th} level, the obtained value of Euclidean distance $4^{u1-v} d^2_{m,v}(I, C_i)$ is discarded completely and then $4^{u1-(v+1)} d^2_{m,v+1}(I, C_i)$ is computed again from the very beginning for next test at the $(v+1)^{th}$ level. This is surly a waste of computation.

3. PROPOSED METHOD

Because fast search for VQ only concerns about finding $d_{\min}$ so as to determine the winner, it has nothing to do with displaying a reconstructed image correctly. Thus, instead of using a 4-PM mean pyramid like [5], [6], we introduce a 2-pixel-merging (2-PM) sum pyramid as shown in Fig.1 (b) in this paper. Since both sum and the mean are L_1 norm in a broad meaning, they must have the same rejection power. Clearly, one advantage of a 2-PM sum pyramid is that sum is exact for operations in integer form. Furthermore, because a pyramid data structure corresponds to the multi-resolution concept in principle, more levels in a pyramid can provide more resolutions for rejection tests and would be profitable to an earlier rejection in VQ encoding. To construct more levels, both 2-PM and 3-PM merging way can be taken into account. Since an image block is conventionally of size $2^x \times 2^x$ for x∈[1~4], only 2-PM but 3-PM merging way is practical for realizing a balanced pyramid. Therefore, another advantage of a 2-PM sum pyramid is that it provides more possibilities to reject a candidate codeword at the cost of more extra memory requirement, which is $[(1+2^1+2^2+ \cdots +2^{u2-1})- (1+4^1+4^2+ \cdots +4^{u1-1})]=(2^{u2+1}-2)/3=(2\times k-2)/3$ more than a 4-PM mean pyramid. For a 4×4 block, it needs 10 more extra memories.

Similar to Eq.3, a new hierarchical rejection rule can be obtained based on the 2-PM sum pyramid data structure as

$$d^2_{s,u2}(I,C_i) \geq \cdots \geq 2^{-(u2-v)} d^2_{s,v}(I,C_i) \geq \cdots \geq 2^{-(u2-1)} d^2_{s,1}(I,C_i) \geq 2^{-u2} d^2_{s,0}(I,C_i) \quad (4)$$

where Euclidean distance at the v^{th} level for v∈[0~u2] is $d^2_{s,v}(I,C_i) = \sum_{m=1}^{2^v} (SI_{v,m} - SC_{i,v,m})^2$; $SI_{v,m}$ is the m^{th} pixel (sum) at the v^{th} level in a 2-PM sum pyramid for I and $SC_{i,v,m}$ implies the same thing to C_i for m∈[1~2^v]. The real Euclidean distance $d(I, C_i)=d_{s,u2}(I, C_i)$ holds. For a 4×4 block, u2=4.

At any v^{th} level for v∈[0~u2], if $2^{-(u2-v)} d^2_{s,v}(I, C_i) > d^2_{\min}$ holds, then the current real Euclidean distance $d^2_{s,u2}(I, C_i)$ will be definitely larger than $d^2_{\min}$. Therefore the search can be terminated at this v^{th} level and C_i can be rejected safely.

Comparing Eq.4 with Eq.3, it is obvious that Eq.4 has doubled the number of tests for a possible rejection. The tests of Eq.4 at the even levels (i.e. v=0, 2, 4, … ,u2) have the same rejection power as Eq.3 but the tests of Eq.4 at the odd levels are newly

inserted. If Eq.4 is directly used, when a test at the v^{th} level fails, the obtained $d^2_{s,v}(I, C_i)$ is also discarded completely and $d^2_{s,v+1}(I, C_i)$ will be computed once again from the very beginning for the following test at $(v+1)^{th}$ level. It is certainly a waste to discard $d^2_{s,v}(I, C_i)$. How to reuse the obtained $d^2_{s,v}(I, C_i)$ to compute $d^2_{s,v+1}(I, C_i)$ recursively is the key of this paper.

Proof of Eq.4

Based on a 2-PM sum pyramid data structure, it is clear that $SI_{v,m} = SI_{v+1,2m-1} + SI_{v+1,2m}$ and $SC_{i,v,m} = SC_{i,v+1,2m-1} + SC_{i,v+1,2m}$ are true for $m \in [1 \sim 2^v]$. By rewriting $d^2_{s,v+1}(I, C_i)$ in its odd and even terms respectively and using $(a^2+b^2) \geq (a+b)^2/2$, we have

$$\begin{aligned}
d^2_{s,v+1}(I, C_i) &= \sum_{m=1}^{2^{v+1}} \left(SI_{v+1,m} - SC_{i,v+1,m}\right)^2 \\
&= \sum_{m=1}^{2^v} \left[\left(SI_{v+1,2m-1} - SC_{i,v+1,2m-1}\right)^2 + \left(SI_{v+1,2m} - SC_{i,v+1,2m}\right)^2\right] \\
&\geq 2^{-1} \sum_{m=1}^{2^v} \left[\left(SI_{v+1,2m-1} - SC_{i,v+1,2m-1}\right) + \left(SI_{v+1,2m} - SC_{i,v+1,2m}\right)\right]^2 \\
&= 2^{-1} \sum_{m=1}^{2^v} \left[\left(SI_{v+1,2m-1} + SI_{v+1,2m}\right) - \left(SC_{i,v+1,2m-1} + SC_{i,v+1,2m}\right)\right]^2 \\
&= 2^{-1} \sum_{m=1}^{2^v} \left[SI_{v,m} - SC_{i,v,m}\right]^2 \\
&= 2^{-1} d^2_{s,v}(I, C_i)
\end{aligned}$$

Because the initial distance computation is $d^2_{s,0}(I, C_i)$ and totally (u2+1) levels are available, Eq.4 can be easily achieved by using the relation $d^2_{s,v+1}(I, C_i) \geq 2^{-1} d^2_{s,v}(I, C_i)$.

A search-efficient special case of 2-PM sum pyramid that only used a rejection test at L_1 level has been proposed in [7].

During a search process, in order to avoid discarding the obtained $d^2_{s,v}(I, C_i)$ value, a recursive way for computing successive $d^2_{s,v+1}(I, C_i)$ is proposed by using $(a^2+b^2)=(a+b)^2-2ab$

$$\begin{aligned}
d^2_{s,v+1}(I, C_i) &= \sum_{m=1}^{2^{v+1}} \left(SI_{v+1,m} - SC_{i,v+1,m}\right)^2 \\
&= \sum_{m=1}^{2^v} \left[\left(SI_{v+1,2m-1} - SC_{i,v+1,2m-1}\right)^2 + \left(SI_{v+1,2m} - SC_{i,v+1,2m}\right)^2\right] \\
&= \sum_{m=1}^{2^v} \left[\left(SI_{v+1,2m-1} - SC_{i,v+1,2m-1}\right) + \left(SI_{v+1,2m} - SC_{i,v+1,2m}\right)\right]^2 \\
&\quad - 2 \times \sum_{m=1}^{2^v} \left[\left(SI_{v+1,2m-1} - SC_{i,v+1,2m-1}\right)\left(SI_{v+1,2m} - SC_{i,v+1,2m}\right)\right] \\
&= \sum_{m=1}^{2^v} \left[\left(SI_{v,m}\right) - \left(SC_{i,v,m}\right)\right]^2 \\
&\quad - 2 \times \sum_{m=1}^{2^v} \left[\left(SI_{v+1,2m-1} - SC_{i,v+1,2m-1}\right)\left(SI_{v+1,2m} - SC_{i,v+1,2m}\right)\right] \\
&= d^2_{s,v}(I, C_i) \\
&\quad - \sum_{m=1}^{2^v} \left[\left(2 \times SI_{v+1,2m-1} - 2 \times SC_{i,v+1,2m-1}\right)\left(SI_{v+1,2m} - SC_{i,v+1,2m}\right)\right]
\end{aligned} \tag{5}$$

From Eq.5, it is obvious that $d^2_{s,v}(I, C_i)$ is completely reused again so that $d^2_{s,v+1}(I, C_i)$ can be computed in a recursive way. Eq.5 is actually a realization of the multi-resolution concept, which implies that any higher resolution can be obtained by just adding some improvements recursively into a lower resolution other than computing it from the very beginning once again. Eq.5 would reduce total computational burden.

Concerning computational complexity, when $d^2_{s,v+1}(I,C_i)$ is computed directly, it needs $2^{v+1}+(2^{v+1}-1) = 2\times 2^{v+1}-1$ addition (±) and 2^{v+1} multiplication (×) operations. However, by storing the doubled odd terms at each level (i.e. to store $2\times SI_{v+1,\,2m-1}$ and $2\times SC_{i,v+1,\,2m-1}$ as a whole instead of the raw $SI_{v+1,2m-1}$ and $SC_{i,v+1,2m-1}$), Eq.5 only needs $[2\times 2^v + (2^v-1)+1] = (3/2)\times 2^{v+1}$ addition (±) and $2^v = (1/2)\times 2^{v+1}$ multiplication (×) operations. Because a multiplication (×) is much heavier than an addition (±) operation, Eq.5 can save about half of the computational burden compared to a direct computation way of $d^2_{s,v+1}(I, C_i)$.

It is clear that 2-PM sum pyramid is an extension of 4-PM pyramid. In order to realize a recursive computation way to reduce more computational burden, only 2-PM sum pyramid can be used but 4-PM pyramid cannot. The reasons is that by using the formula $a^2+b^2=(a+b)^2-2ab$, once multiplication (×) can be saved when the value of $(a+b)^2$ is known. In contrast, the formula $a^2+b^2+c^2+d^2=(a+b+c+d)^2-2ab-2ac-2ad-2bc-2bd-2cd$ cannot save any computational burden even though the value of $(a+b+c+d)^2$ is known. Therefore, 2-PM sum pyramid is a very promising data structure for fast VQ search.

Based on the discussions above, a search flow can be summarized as follows: (1) Construct an accompanying 2-PM sum pyramid for each C_i off-line. Double the values of odd terms for $v \in [1 \sim u2]$ levels instead of their original values so as to take Eq.5 into account. (2) Sort all codewords by the real sum at L_0 level in ascending order off-line. (3) For an input I, construct its accompanying 2-PM sum pyramid on-line. Similarly, double the values of odd terms for $v \in [1 \sim u2]$ levels. (4) Find an initial nearest (NN) codeword C_N among the sorted codewords by a binary search, which is the closest codeword in terms of real sum difference $d_{s,0}(I, C_N) = \left|SI_{0,1} - SC_{N,0,1}\right|$ being minimum. It needs $\log_2(N_c)$ times comparisons (cmp). Then compute and store "so far" $d^2_{\min} = d^2(I, C_N)$, and $d^2_{v,\min} = 2^{u2-v} \times d^2_{\min}$ to simplify a future test at the v^{th} level, $v \in [0 \sim u2-1]$. This step needs $(2\times k-1)$ additions (±) and $[k+\log_2(n\times n)]$ multiplications (×). (5.1) Continue the winner search up and down around C_N one by one. Once $\left(SI_{0,1} - SC_{i,0,1}\right)^2 \geq d^2_{0,\min}$ holds, terminate search for the upper part of sorted codebook when $i<N$ or the lower part when $i>N$; If winner search in both upper and lower directions has been terminated, search is complete. This step is actually an application of ENNS method. Clearly, the current "so far" best-matched codeword must be winner. Then search flow returns to Step 3 for encoding another new input. (5.2) Otherwise, test whether $d^2_{s,v}(I, C_i) \geq d^2_{v,\min}$ is true or not for $v \in [1 \sim u2]$. If it is true, reject C_i safely. (Note, Eq.5 must be introduced here for computing $d^2_{s,v}(I, C_i)$ in a recursive way.) (6) If all tests fail for a rejection, it implies that current C_i is a better-matched codeword, then update $d^2_{\min}$ by $d^2_{s,u2}(I, C_i)$ and all $d^2_{v,\min}$ for $v \in [0 \sim u2-1]$. Meanwhile, update the winner index "so far". Then, return to Step (5.1) to check next codeword.

4. EXPERIMENTAL RESULTS

Simulation experiments with MATLAB are conducted. Codebooks are generated by using a modified Kohonen's self-organizing map (SOM) method [8] with the 512×512 8-bit Lena image as a training set. Block size is 4×4. Two kinds of evaluations are made. The first evaluation is how small the search space can be reduced per input vector after all rejection steps completed in each method. For EEENNS method, it is the remaining Euclidean distance computations after ENNS, EENNS and EEENNS rejection tests; for 4-PM mean pyramid method, it is those after rejection tests at $L_0 \sim L_{u1-1}$ levels; and for 2-PM sum pyramid method, it is those after rejection tests at $L_0 \sim L_{u2-1}$ levels. The second evaluation is the total computational burden including all overhead in the number of addition (±),

multiplication (×) and comparison (cmp) operations per input vector, which consists of (1) finding the initial best-matched codeword C_N and computing the initial d^2_{min}, $d^2_{v,min}$; (2) computing test condition using each statistical feature or each level in pyramids for a possible rejection before bottom level and updating the "so far" d^2_{min}, $d^2_{v,min}$; (3) computing the remaining Euclidean distances. When k=16, 1 Euclidean distance computation equals 31 additions (±), 16 multiplications (×) and 1 comparison (cmp).

The results of first evaluation are summarized in TABLE 1.

TABLE 1
COMPARISON OF REDUCED SEARCH SPACE OR RMAINING EUCLIDEAN DISTANCE COMPUTATIONS PER INPUT VECTOR AFTER ALL REJECTION TESTS COMPLETED

Size	Method	Lena	F-16	Pepper	Baboon
256	FS	256	256	256	256
	EEENNS	13.62	13.29	15.77	46.64
	4-PM mean	8.71	6.96	9.37	25.12
	2-PM sum	3.71	2.99	3.89	8.68
512	FS	512	512	512	512
	EEENNS	23.84	24.6	29.57	91.47
	4-PM mean	13.54	11.7	16.05	47.54
	2-PM sum	4.69	3.97	5.3	15.03
1024	FS	1024	1024	1024	1024
	EEENNS	37.56	45.13	52.92	174.6
	4-PM mean	19.12	19.1	26.29	82.41
	2-PM sum	5.61	5.19	7.14	21.89

From TABLE 1, it is obvious that much more search space can be reduced by using 2-PM sum pyramid data structure. The key reason is that its L_{u2-1} level has a very high resolution so that it can reject most of candidate codewords before a real Euclidean distance computation.

Taking codebook size=256 as an example, the results of second evaluation is summarized in TABLE 2.

TABLE 2
COMPARISON OF TOTAL COMPUTATIONAL BURDEN PER INPUT VECTOR

Method	Operation	Lena	F-16	Pepper	Baboon
FS	Add	7936	7936	7936	7936
	Mul	4096	4096	4096	4096
	CMP	256	256	256	256
EEENNS	Add	513.6	493.9	591.4	1706.4
	Mul	289.9	277.7	333.4	953.5
	CMP	72.9	67.9	81.9	208.1
4-PM mean	Add	424.3	353.1	462.8	1199.3
	Mul	235.5	197.1	257.6	664.7
	CMP	54.1	48.3	59.8	137.6
2-PM sum	Add	319.4	265.4	348.9	919.9
	Mul	124.9	105.2	136.3	347.3
	CMP	66.3	58.1	72.9	179.4

From TABLE 2, it is also obvious that the total computational burden by using a 2-PM sum pyramid data structure is much less compared to other previous works, especially the number of multiplication operations can be reduced greatly. This mainly benefits from computing $d^2_{s,v}(I, C_i)$ by using Eq.5 for $v \in [1 \sim u2]$ in a recursive way.

Of course, the cost for the 2-PM sum pyramid method is that it needs more extra memories. Because the purpose of a fast search method for VQ is its encoding speed, if the extra memory requirement is allowed, the 2-PM sum pyramid method in a recursive way is very promising and practical.

5. CONCLUSION

In this paper, a 2-PM sum pyramid data structure and a recursive computation way for Euclidean distance at each level are proposed to improve the performance of a conventional 4-PM mean pyramid for fast VQ encoding. Three advantages are realized. First, sum is used to replace the mean so as to avoid roundoff error problem. Second, another new level is inserted in-between every two neighboring levels in a conventional 4-PM pyramid, which corresponds to an additional resolution and would be profitable to enhance the capability of multi-resolution processing for rejection tests. Third, a recursive way for computing Euclidean distance at each level is proved in order to save computational burden significantly, which is the most important point in this paper because it exploits the potential power of multi-resolution processing thoroughly. In fact, it is unnecessary to discard any obtained value of Euclidean distance at any level so as to avoid the waste of computation. Experimental results confirmed that the proposed method outperforms the previous works obviously.

6. REFERENCES

[1] N.M.Nasarabadi and R.A.King, "Image coding using vector quantization: A review," IEEE Trans. Commun., vol. 36, pp.957-971, Aug. 1988.

[2] L.Guan and M.Kamel, "Equal-average hyperplane partitioning method for vector quantization of image data," Pattern Recognition Letters, vol.13, pp.693-699, Oct. 1992.

[3] S.Baek, B.Jeon and K.Sung, "A fast encoding algorithm for vector quantization," IEEE Signal Processing Letters, vol.4, pp.325-327, Dec. 1997.

[4] Z.M.Lu and S.H.Sun, "Equal-average equal-variance equal-norm nearest neighbor search algorithm for vector quantization," IEICE Tans. Information and System, vol.86D, pp.660-663, March 2003.

[5] C.H.Lee and L.H.Chen, "A fast search algorithm for vector quantization using mean pyramids of codewords," IEEE Trans. Commun., vol.43, pp.1697-1702, Feb. 1995.

[6] S.J.Lin, K.L.Chung and L.C.Chang, "An improved search algorithm for vector quantization using mean pyramid structure," Pattern Recognition Letters, vol.22, pp.373-379, 2001.

[7] Z.Pan, K.Kotani and T.Ohmi, "A hierarchical fast encoding algorithm for vector quantization with PSNR equivalent to full search," IEEE International Symposium on Circuits and Systems (ISCAS2002), vol.(I), pp.797-800, 2002.

[8] T.Nozawa, M.Konda, M.Fujibayashi, M.Imai, K.Kotani, S.Sugawa and T.Ohmi, "A parallel vector-quantization processor eliminating redundant calculations for real-time motion picture compression," IEEE J. Solid-State Circuits, vol.35, pp.1744-1751, Nov. 2000.