FAST ENCODING METHOD FOR VECTOR QUANTIZATION BASED ON A NEW MIXED PYRAMID DATA STRUCTURE

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

VQ is a famous signal compression method. The encoding speed of VQ is a key problem for its practical application. In principle, the high dimension of a vector makes it very expensive computationally to find the best-matched template in a codebook for an input vector by Euclidean distance. As a result, many fast search methods have been developed in previous works based on statistical features (i.e. mean, variance or L_2 norm) or multi-resolution representation (i.e. various pyramid data structures) of a vector to deal with this computational complexity problem. Therefore, how to use them optimally in terms of a small memory requirement and a little computational overhead becomes very important.

This paper proposes to combine both 2-PM sum pyramid and (n×n)-PM variance pyramid of a vector to construct a new mixed pyramid data structure, which only requires (k+1) memories for a k-dimensional vector. Experimental results confirmed that the encoding efficiency by using this mixed pyramid outperforms the previous works obviously.

1. INTRODUCTION

In a vector quantization (VQ) [1] framework, its encoding process is implemented block by block sequentially. The distortion between an n×n input 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. Due to a high dimension k, it is very expensive to compute $d^2(I, C_i)$.

Then, a best-matched codeword (winner) with minimum distortion 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 the winner and its subscript "w" is the winner index. This process for finding the winner is called a full search (FS). Once "w" has been found, which uses much less bits than C_w, VQ only transmits this index "w" instead of C_w to reduce data amount for image compression. Because the same codebook has also been stored at the receiver, by using the received index "w" to retrieval C_w at the receiver, it is very easy to reconstruct the image.

Obviously, the principle of VQ encoding implies that only the sole winner C_w has to be found by an exact Euclidean distance computation but all other C_i (i≠w) has to be rejected actually. In other words, VQ encoding can also be viewed as a process for rejecting all non-best-matched codewords rather than finding a best-matched codeword. This property of VQ provides a possibility of estimating real Euclidean distance by an inexpensive computation to see whether it is really "large" enough so as to make a codeword rejection.

Clearly, in order to high-efficiently estimate real Euclidean distance, the key issue is how to construct appropriate features or a pyramid data structure for a k-dimensional vector. This paper aims at enhancing 2-PM sum pyramid [8] by mixing it with a special (n×n)-PM variance pyramid of a vector to improve the search efficiency further.

2. RELATED PREVIOUS WORKS

Because a k-dimensional vector can be equivalently viewed as a k-sample set, it is clear that the mean, the variance and L_2 norm are its statistical features. They can be directly used to measure how different two vectors are. During a winner search process, suppose the "so far" minimum Euclidean distance is d_{min}. Based on the mean information, the previous work [2] proposed a rejection rule as: If $k\left(MI - MC_i\right)^2 \geq d^2_{\min}$ 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 rejection rule when ENNS method fails as: If $k(MI - MC_i)^2 + (VI - VC_i)^2 \geq d^2_{\min}$ 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 additional codeword rejection rule when EENNS method also fails as: If $\left(L_2 I - L_2 C_i\right)^2 \geq d^2_{\min}$ holds, then reject C_i safely as well, where L_2I is the L_2 norm of I and L_2C_i means the same for C_i. This is EEENNS method (i.e. equal-average equal-variance equal-norm nearest neighbor search). EEENNS method is the current fastest search method by only using the statistical features of a vector. EEENNS method needs three extra memories for storing the mean, the variance and L_2 norm for each C_i. However, the previous work [8] showed that EEENNS method performs poor compared to the fast search methods using pyramid data structures.

On the other hand, in order to realize a multi-resolution instead of a simple statistical feature description for a vector, a pyramid data structure can be taken into account naturally. Then, a 4-pixel-merging (4-PM) mean pyramid as shown in Fig.1 (a) is proposed in the previous works [5], [6].

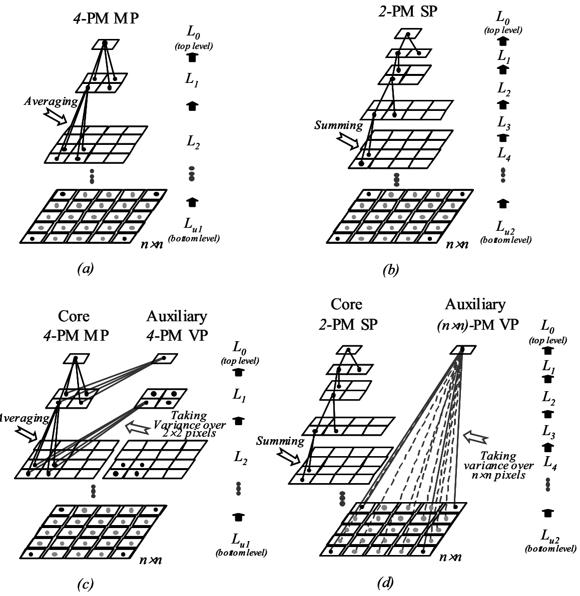

Fig.1. For an n×n block, (a) a 4-PM mean pyramid (MP) with bottom level $u1=\log_4(n\times n)$; (b) a 2-PM sum pyramid (SP) with bottom level $u2=\log_2(n\times n)=2\times u1$; (c) a 4-PM mean-variance pyramid (MP+VP) with bottom level $u1=\log_4(n\times n)$; and (d) a mixed pyramid with bottom level $u2=\log_2(n\times n)=2\times u1$.

A hierarchical rejection rule is set up in [5], [6] as

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

Suppose $MI_{v,m}$ is the m^{th} pixel at the v^{th} level in a 4-PM mean pyramid for I and $MI_{v,m}=(MI_{v+1,4m-3}+MI_{v+1,4m-2}+MI_{v,+1\,4m-1}+MI_{v+1,4m})/4$ for $m\in[1\sim4^v]$. $MC_{i,v,m}$ implies the same thing to C_i. Then Euclidean distance at the v^{th} level for $v\in[0\sim u1]$ between the two 4-PM mean pyramids of I and C_i is $d^2_{m,v}(I,C_i)\overset{Def}{=}\sum_{m=1}^{4^v}(MI_{v,m}-MC_{i,v,m})^2$. Because $d_{m,u1}(I, C_i) = d(I, C_i)$, it is called as real Euclidean distance for the bottom level L_{u1}. For a 4×4 block, u1=2.

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

In order to realize more resolutions by a pyramid data structure and meanwhile realize recursive computation in a memory-efficient way, a 2-PM sum pyramid as shown in Fig.1 (b) is proposed in [8]. A hierarchical rejection rule is set up as

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

where Euclidean distance at the v^{th} level for $v\in[0\sim u2]$ is $d^2_{s,v}(I,C_i)\overset{Def}{=}\sum_{m=1}^{2^v}(SI_{v,m}-SC_{i,v,m})^2$, $SI_{v,m}$ is the m^{th} pixel at the v^{th} level for I and $SI_{v,m}=(SI_{v+1,2m-1}+SI_{v+1,2m})$ for $m\in[1\sim2^v]$. $SC_{i,v,m}$ means the same to C_i. The real Euclidean distance $d^2_{s,u2}(I, C_i)=d^2(I, C_i)$ holds. For a 4×4 block, u2=4.

Thus, at any v^{th} level for $v\in[0\sim u2]$, if $2^{-(u2-v)}d^2_{s,v}(I, C_i)>d^2_{min}$ holds, then the current real Euclidean distance $d^2_{s,u2}(I, C_i)$ is definitely larger than d^2_{min}. As a result, the search can be terminated at this v^{th} level and C_i can be rejected safely.

Here, the Euclidean distance at $(v+1)^{th}$ level for $v\in[0\sim u2-1]$ must be computed in the following recursive way

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

As interpreted in [8], this 2-PM sum pyramid only requires n×n memories for an n×n block and it can reduce the computational burden to about half compared to the conventional 4-PM mean pyramid. It is a very practical data structure for fast VQ encoding.

Actually, in addition to mean or sum, variance is also an effective statistical feature of a vector. Let P=(1,1, … ,1) in R^k space be a projection axis in R^k space, which is called as a central axis in [3]. From a viewpoint of the geometry, sum is the projection component of a vector onto P and variance is the orthogonal component from this vector to sum [10]. In other words, sum and variance of a vector can be viewed as an orthogonal decomposition for a vector on the axis P. Because sum and variance are orthogonal in R^k space, it is beneficial to combine them into the pyramid data structure.

A straightforward combining way is proposed in [7] as shown in Fig.1 (c). For an n×n block, two pyramids are constructed over it. The core 4-PM mean pyramid is the same as that in Fig.1 (a) but an auxiliary 4-PM variance pyramid is constructed as well. Suppose $VI_{v,m}$ is the m^{th} pixel at the v^{th} level in a 4-PM variance pyramid for I and $VI_{v,m}=\sqrt{(MI_{v+1,4m-3}-MI_{v,m})^2+(MI_{v+1,4m-2}-MI_{v,m})^2+(MI_{v+1,4m-1}-MI_{v,m})^2+(MI_{v+1,4m}-MI_{v,m})^2}$ for $m\in[1\sim4^v]$, $v\in[0\sim u1-1]$. At the bottom level, $VI_{u1,m}=0$ for $m\in[1\sim4^{u1}]$. $VC_{i,v,m}$ implies the same thing to C_i. Then Euclidean distance at the v^{th} level for $v\in[0\sim u1]$ between the two 4-PM variance pyramids is $d^2_{var,v}(I,C_i)\overset{Def}{=}\sum_{m=1}^{4^v}(VI_{v,m}-VC_{i,v,m})^2$. For a 4×4 block, u1=2. Therefore, a composite Euclidean distance can be defined as $D^2_{mv,v}(I,C_i)\overset{Def}{=}d^2_{m,v}(I,C_i)+d^2_{var,v}(I,C_i)/4$. Then, a hierarchical rejection rule that is similar to Eq.3 is set up in [7] as

$$d^2(I,C_i)=D^2_{mv,u1}(I,C_i)\ge\cdots\ge 4^{u1-v}D^2_{mv,v}(I,C_i)\ge \cdots\ge 4^{u1-1}D^2_{mv,1}(I,C_i)\ge 4^{u1}D^2_{mv,0}(I,C_i) \quad (6)$$

Because $D^2_{mv,v}(I,C_i)\ge d^2_{m,v}(I,C_i)$ is always true, it is clear that Eq.6 is more powerful for rejection than Eq.3.

However, Eq.6 will certainly introduce much more memory overhead and computational overhead compared to Eq.3. First, a 4-PM mean-variance pyramid doubles the memory requirement for each codeword. And unlike the mean feature, the computation of variance is non-linear so that it is difficult to reduce this memory requirement by developing a memory-efficient storing way such as 2-PM sum pyramid in [8]. Second, because the input vector is unknown before encoding, its 4-PM variance pyramid must be constructed on-line, which will totally need $(4^{u1-1}+\ldots+4^2+4^1+1)\times7=(n\times n-1)/3\times7$ additions (±), $(4^{u1-1}+\ldots+4^2+4^1+1)\times4=(n\times n-1)/3\times4$ multiplications (×) and $(4^{u1-1}+\ldots+4^2+4^1+1)\times1=(n\times n-1)/3\times1$ square root operations (sqrt). For a 4×4 block, it needs 35 "±", 20 "×" and 5 "sqrt" operations. This is a large extra computational overhead. Third, the computational cost of $D^2_{mv,v}(I, C_i)$ at each level also doubles that of $d^2_{m,v}(I, C_i)$ so that it will also introduce a lot of overhead. From the experimental results in [7], 4-PM mean-variance pyramid can improve the overall search efficiency by 15%~20% compared to 4-PM mean pyramid.

3. PROPOSED METHOD

From the analysis above, it concludes that it is rather effective by combining an auxiliary 4-PM variance pyramid with a core 4-PM mean pyramid at the cost of a large memory and computational overhead. Therefore, how to improve these two pyramids further becomes very important. In order to enhance the core 4-PM mean pyramid, [8] proposed a promising 2-PM sum pyramid as shown in Fig.1 (b) to compute Euclidean distance recursively, in which way no extra memory requirement is needed and about half of the total computational burden can be reduced. In contrast, 4-PM mean pyramid cannot realize recursive computation at all in principle [8]. Therefore, it concludes that 2-PM sum pyramid should be adopted as a core pyramid. However, it is impossible to directly combine it with a 2-PM variance pyramid because of too much overheads.

Clearly, in order to reduce the overheads introduced by an auxiliary variance pyramid, it is only possible to make the variance pyramid "smaller", which implies that it only has very few levels in it. The reason is that variance pyramid is constructed by non-linear operations. The special case is to construct a (n×n)-PM auxiliary variance pyramid as shown in Fig.1 (d), which only has the top level but without any intermediate levels (Note: the value at top level here is usually different from that at top level in a 4-PM variance pyramid as shown in Fig.1 (b)). In Fig.1 (d), because the core 2-PM sum pyramid is completed (i.e. with all intermediate levels) but the (n×n)-PM variance pyramid is incomplete, it is called as a mixed pyramid due to this asymmetry .

To use the mixed pyramid, firstly, it only requires one extra memory to store auxiliary (n×n)-PM variance pyramid for each codeword. It is a very small memory requirement. Secondly, for an input block, it only requires (2×n×n−1) additions (±), (n×n) multiplications (×) and one square root operations (sqrt) for on-line constructing its pyramid. For a 4×4 block, it needs 31 "±", 16 "×" and 1 "sqrt" operations. This is a small extra computational overhead. Especially, "sqrt" operation becomes much less. Thirdly, because only the rejection test at the top level is changed, it will not introduce any computational overhead for rejection tests at other remaining levels like [7]. Because the rejection tests are executed from the top level, this mixed pyramid would be more powerful for rejection than 2-PM sum pyramid.

Based on Eq.4, a new hierarchical rejection rule can be set up as

$$d^2(I,C_i)=d^2_{s,u2}(I,C_i)\ge\cdots\ge 2^{-(u2-v)}d^2_{s,v}(I,C_i)\ge \cdots\ge 2^{-(u2-1)}d^2_{s,1}(I,C_i) \;\Uparrow\; \ge 2^{-u2}d^2_{s,0}(I,C_i) \quad (7)$$

Insert a new rejection test:

$$2^{-u2}d^2_{s,0}(I,C_i)+(VI-VC_i)^2\ge$$

where VI is the variance of I and VC_i means the same for C_i.

Obviously, Eq.7 holds because the new inserted test is equivalent to EENNS method but mixed with a core 2-PM sum pyramid. From the concept of pyramid data structure, it is clear that a mixed pyramid in Fig.1 (d) is the minimum configuration by using a "so far" most efficient core 2-PM sum pyramid that has a maximum number of levels for an n×n block and a "smallest" auxiliary (n×n)-PM variance pyramid that guarantees the least memory overhead and rather less computational overhead. This mixed pyramid is a more promising data structure for fast VQ encoding.

Based on the discussions above, a search flow can be summarized as follows: (1) Construct an accompanying core 2-PM sum pyramid and an auxiliary (n×n)-PM variance pyramid for each C_i off-line. For the core 2-PM sum pyramid, only store the odd terms but the even terms for v∈[0~u2] levels. It needs (n×n+1) memories in total for each codeword. (2) Sort and rearrange all codewords by the real sum at L_0 level in ascending order off-line. (3) For an input I, construct its accompanying core 2-PM sum pyramid and the auxiliary (n×n)-PM variance pyramid on-line. Similarly, for the core 2-PM sum pyramid only store its odd terms but the even terms for v∈[0~u2] levels. (4) Find an initial nearest (NN) codeword C_N among the sorted codewords by using a binary search, which is the closest codeword in terms of real sum difference $d_{s,0}(I,C_N)=|SI_{0,1}-SC_{N,0,1}|$ being minimum. It needs $\log_2(N_c)$ times comparisons (cmp). Then compute and temporally store "so far" $d^2_{\min}=d^2(I,C_N)$, $d^2_{v,\min}=2^{u2-v}\times d^2_{\min}$ in order to simplify a future rejection test at the v^{th} level for v∈[0~u2−1]. This step needs (2k−1) additions (±) and (k+u2) multiplications (×) operations. (5.1) Continue the winner search up and down around initial NN C_N one by one. Rejection tests for each candidate codeword are executed from the top level towards the bottom level. At the top level, once $2^{-u2}d^2_{s,0}(I,C_i)+(VI-VC_i)^2\ge d^2_{\min}$ 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. Clearly, the current "so far" best-matched codeword must be the winner. Then search flow returns to Step 3 for encoding another new input; (5.2) Else, for the remaining v∈[1~u2] levels, test whether $d^2_{s,v}(I,C_i)\ge d^2_{v,\min}$ is true or not. If it is true, reject C_i safely. (Note, Eq.5 must be introduced here for computing $d^2_{s,v}(I, C_i)$ recursively.) (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∈[0~u2−1] again. Meanwhile, update the corresponding winner index "so far". Then, return to Step (5.1) to test next codeword.

4. EXPERIMENTAL RESULTS

To compare the search performance with previous works, simulation experiments using MATLAB are conducted. Codebooks of size 256, 512 and 1024 are generated using 512×512, 8-bit Lena image as a training set based on [10]. Block size is 4×4. Because [8] has shown that 2-PM sum pyramid method outperforms EEENNS method, 4-PM mean pyramid method and [7] has shown that 4-PM mean-variance pyramid method outperforms 4-PM mean pyramid method, the performance comparisons are just made among 4-PM mean-variance pyramid, 2-PM sum pyramid and mixed pyramid method in this paper.

Search efficiency is evaluated by total computational burden in terms of the number of addition (±), multiplication (×), comparison (cmp) and square root (sqrt) operations per input vector, which consists of (1) on-line constructing the mixed pyramid for an input vector I; (2) finding the initial best-matched codeword C_N and computing the initial $d^2_{\min}$,

$d^2_{v,min}$; (3) computing test condition for a possible rejection at each intermediate level in a pyramid method; (4) computing the real Euclidean distance at the bottom level and updating "so far" d^2_{min}, $d^2_{v,min}$ again if current codeword is a better-matched one. The results are summarized in Table 1.

From Table 1, it is clear that the mixed pyramid method achieves the best search performance, especially the number of multiplication (×) operations can be reduced greatly. The reason is that it has a more powerful top level for rejection test compared to the 2-PM sum pyramid method. The reason is also that it can benefit from a high-efficient recursive computation using Eq.5 for rejection tests at all intermediate levels compared to a 4-PM mean-variance pyramid method that uses composite $D^2_{mv,v}(I, C_i)$ in Eq.6.

5. CONCLUSION

In this paper, a practical new mixed pyramid is proposed that can obviously improve the search efficiency of the previous works. Two issues are made clear. First, a pyramid data structure should be constructed by using both a core mean (sum) pyramid and an auxiliary variance pyramid. The reason is that the mean (sum) and the variance component of a vector are orthogonal in R^k space. Second, it is profitable to use the high-efficient 2-PM sum pyramid as a core pyramid because it can thoroughly exploit the linear property contained in itself to realize a recursive computation for Euclidean distance and provide a maximum number of levels. Meanwhile, it is also profitable to use (n×n)-PM variance pyramid as an auxiliary variance pyramid because it just has a minimum number of levels so as to avoid the large overhead introduced by the non-linear property contained in itself. Actually, the mixed pyramid in this paper is consisted of a "largest" core sum pyramid and a "smallest" auxiliary variance pyramid over an n×n block, which is a more reasonable configuration for pyramid data structure.

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 Trans. Information and System*, vol.E86-D, 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] J.S. Pan, Z.M Lu and S.H. Sun, "Fast codeword search algorithm for image codeing based on mean-variance pyramids of codewords," *Electronics Letters*, vol.36, pp.210-211, Feb. 2000.

[8] Z.Pan, K.Kotani and T.Ohmi, "A memory-efficient fast encoding method for vector quantization using 2-pixel-merging sum pyramid," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2004)*, III-669-672, 2004.

[9] Z.Pan, K.Kotani and T.Ohmi, "A unified projection method for fast search of vector quantization," *IEEE Signal Processing Letters*, Vol. 11, pp.637-640, July 2004.

[10] 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.

TABLE 1

COMPARISON OF TOTAL COMPUTATIONAL BURDEN PER INPUT VECTOR

Size	Method	Operation	Lena	F-16	Pepper	Baboon
256	Full search	add	7936	7936	7936	7936
		mul	4096	4096	4096	4096
		cmp	256	256	256	256
		sqrt	0	0	0	0
	4-PM man-variance pyramid	add	381.2	317.8	399.3	1127.8
		mul	205.3	171.8	216.0	608.2
		cmp	37.9	33.2	41.1	105.2
		sqrt	5	5	5	5
	2-PM sum pyramid	add	358.5	297.8	392.0	1038.9
		mul	125.1	105.5	136.4	347.5
		cmp	66.0	58.0	72.7	179.2
		sqrt	0	0	0	0
	Mixed pyramid	add	300.6	253.5	314.4	865.6
		mul	112.9	97.3	118.2	300.5
		cmp	52.6	46.0	57.5	154.9
		sqrt	1	1	1	1
512	Full search	add	15872	15872	15872	15872
		mul	8192	8192	8192	8192
		cmp	512	512	512	512
		sqrt	0	0	0	0
	4-PM man-variance pyramid	add	583.6	506.2	662.0	2049.1
		mul	317.0	276.3	361.0	1110.2
		cmp	61.5	55.9	71.2	197.2
		sqrt	5	5	5	5
	2-PM sum pyramid	add	556.9	497.2	661.9	1968.0
		mul	191.7	172.2	226.4	652.6
		cmp	106.1	97.8	125.1	340.0
		sqrt	0	0	0	0
	Mixed pyramid	add	445.8	390.1	501.5	1569.4
		mul	163.6	145.3	183.5	538.0
		cmp	87.3	79.5	101.7	293.4
		sqrt	1	1	1	1
1024	Full search	add	31744	31744	31744	31744
		mul	16384	16384	16384	16384
		cmp	1024	1024	1024	1024
		sqrt	0	0	0	0
	4-PM man-variance pyramid	add	847.4	838.4	1088.1	3684.8
		mul	464.9	461.5	598.8	2005.2
		cmp	96.1	97.2	124.3	365.6
		sqrt	5	5	5	5
	2-PM sum pyramid	add	820.3	826.0	1104.8	3450.2
		mul	282.1	283.7	376.3	1144.4
		cmp	166.2	168.7	218.4	623.0
		sqrt	0	0	0	0
	Mixed pyramid	add	637.6	625.7	806.1	2766.1
		mul	232.7	228.9	292.0	944.5
		cmp	139.7	140.6	180.8	546.6
		sqrt	1	1	1	1