A Unified Projection Method for Fast Search of Vector Quantization

著者	小谷　光司
journal or publication title	IEEE signal processing letters
volume	11
number	7
page range	637-640
year	2004
URL	http://hdl.handle.net/10097/34916

A Unified Projection Method for Fast Search of Vector Quantization

Zhibin Pan, *Member, IEEE*, Koji Kotani, *Member, IEEE*, and Tadahiro Ohmi, *Fellow, IEEE*

***Abstract*—Vector quantization (VQ) is a famous asymmetric signal compression method. In VQ, the search process to find the winner for an input vector is extremely time consuming due to a lot of k-dimensional Euclidean distance computations. This property of VQ constrains its practical applications to some extent. In order to speed up the search process of VQ, a unified projection method is proposed in this letter to reject a candidate code vector by a lighter computational burden. This method is universal because it can unify several types of previous works through suitably selecting a projection axis. Furthermore, two criteria for how to select an optimal projection axis for a code vector are proven mathematically, which are most important because they demonstrate the direction for a potential improvement to the search efficiency of VQ.**

Experimental results of VQ encoding show that the proposed method is very search effective.

***Index Terms*—Euclidean distance estimation, fast search, unified projection method, vector quantization (VQ).**

I. Introduction

Vector quantization (VQ) [1] is a classical but still very promising method for signal compression. In order to use VQ, a codebook is necessary from the very beginning and it can usually be generated by the famous LBG [2] method. Once a codebook is available, encoding by VQ is possible. The search process to find the best-matched item (winner) within the codebook for a specific input vector is the time bottleneck of VQ because it needs a lot of Euclidean distance computations.

For an input vector $I = (I_1, I_2, \ldots, I_k)$ and a code vector $C_i = (C_{i,1}, C_{i,2}, \ldots, C_{i,k})$, which is in the codebook $C = \{C_i | i = 1 \sim N_c\}$ of size N_c, the real distortion between I and C_i is actually a difference vector given as $D_i = I - C_i = (D_{i,1}, D_{i,2}, \ldots, D_{i,k})$. For simplicity, this difference is usually measured by squared Euclidean distance in VQ by using either the distance definition or the law of cosines as shown in Fig. 1

$$\begin{aligned} d^2(I, C_i) &= \|D_i\|^2 = \|I - C_i\|^2 = \sum_{j=1}^{k} (I_j - C_{i,j})^2 \\ &= \|I\|^2 + \|C_i\|^2 - 2\|I\|\|C_i\| \cos(\theta) \end{aligned} \quad (1)$$

where $\theta = \cos^{-1}[\langle I, C_i \rangle / (\|I\|\|C_i\|)]$ can be defined as a coupling angle between the intersected I and C_i in R^k space mathematically. Because I and C_i are positive for image data, $\cos(\theta) = \langle I, C_i \rangle / (\|I\|\|C_i\|) \geq 0$ holds, where $\langle\rangle$ means inner product. Hence, $|\theta| \leq 90^\circ$ is true.

Thus, the winner can be determined straightforwardly by

$$d^2(I, C_w) = \min_{C_i \in C} [d^2(I, C_i)] \quad (2)$$

where C_w is the winner. This is called a full search (FS) over the whole codebook. FS is extremely time consuming due to it performing $\mathrm{N_c}$ times k-dimensional Euclidean distance computations. If part of the $\mathrm{N_c}$ times distance computations can be avoided by using some lower dimensional computations instead but exactly the same winner can be guaranteed, the method is defined as FS-equivalent. Many FS-equivalent fast search methods have already been developed to find the winner [3]–[9]. FS-equivalent methods can be applied to both codebook generation and VQ encoding in the same way. This letter will deal with VQ encoding only.

II. Proposed Fast Search Method

To find the winner in VQ, only the final minimum Euclidean distance $d^2(I, C_w)$ is necessary to be exactly computed. This potentially provides a possibility for roughly computing other Euclidean distances or just making estimations for them so as to make a rejection. During a search process, suppose the minimum Euclidean distance found "so far" be $d^2_{\min}$. If current $d^2(I, C_i)$ can be evaluated to be larger than $d^2_{\min}$ by a rough but lighter computation, then C_i can be rejected safely. In this way, the exact but heavier $d^2(I, C_i)$ computation can be avoided.

The heavy computational burden of VQ mainly comes from a high dimension of vectors. To make the dimension of a vector lower, the projection method is a natural consideration. In projection way, a vector can be expressed by a projection component and a corresponding orthogonal component, which implies a one-to-one mapping from a R^k space to a R^2 space. Then it becomes possible to roughly measure the difference between two vectors in this R^2 space. There exist two projection ways for VQ search when considering the physical or geometrical meaning of the difference between two vectors as illustrated in Fig. 1. The first way is to individually project I and C_i onto a projection axis to see whether their difference after the projection is sufficiently large to make a rejection or not. The second way is to directly project D_i to see whether its projection component is large enough to make a rejection or not. Thus, a projection axis has to be set up. Let it be $P = (P_1, P_2, \ldots, P_k)$.

In the first way, three vectors I, C_i, and P intersect at the origin $O = (0, 0, \ldots, 0)$ and they constitute a tri-hedral angle in R^k space. Let θ_1 be the plane angle between (I, P) and θ_2

Manuscript received July 16, 2003; revised October 9, 2003. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Scott T. Acton.

Z. Pan and T. Ohmi are with the New Industry Creation Hatchery Center, Tohoku University, Sendai 980-8579, Japan (e-mail: pzb@fff.niche.tohoku.ac.jp).

K. Kotani is with the Department of Electronic Engineering, Graduate School of Engineering, Tohoku University, Sendai 980-8579, Japan.

Digital Object Identifier 10.1109/LSP.2004.827955

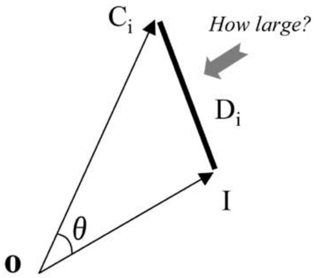

Fig. 1. Concept of the difference vector D_i and the coupling angle θ.

be the plane angle between (C_i, P). According to the *tri-hedral theorem* in solid geometry that states "of the three plane angles that form a tri-hedral angle, any two are together greater than the third," it is clear that both $\theta+\theta_1 \geq \theta_2$ and $\theta+\theta_2 \geq \theta_1$ hold. Thus, $\theta \geq |\theta_1 - \theta_2|$ is identically correct.

Because the cosine function is an even and monotonously decreasing function within $[-90^\circ, 90^\circ]$ interval, based on (1) that uses the law of cosines, we have

$$\begin{aligned} d^2(I, C_i) &\geq \|I\|^2 + \|C_i\|^2 - 2\|I\|\|C_i\| \cos(|\theta_1 - \theta_2|) \\ &= \|I\|^2 + \|C_i\|^2 - 2\|I\|\|C_i\|[\cos(\theta_1)\cos(\theta_2) \\ &\quad + \sin(\theta_1)\sin(\theta_2)]. \end{aligned} \tag{3}$$

On the other hand, based on the concept of vector projection, for (I, P) pair (forming a plane), we have $I_P = \|I\|\cos(\theta_1)$ and $I_{P,\perp} = \|I\|\sin(\theta_1)$, where I_P is the projection of I onto P, and $I_{P,\perp}$ is its sole orthogonal component. For (C_i, P) pair, $(C_i)_P = \|C_i\|\cos(\theta_2)$ and $(C_i)_{P,\perp} = \|C_i\|\sin(\theta_2)$ have the same meaning. Furthermore, any vector w can be expanded with an arbitrary P as a projection axis in an orthogonal way as $\|w\|^2 = (w_P)^2 + (w_{P,\perp})^2$. Thus, (3) can be rewritten as

$$\begin{aligned} &d^2(I, C_i) \\ &\quad \geq \left[I_P^2 + I_{P,\perp}^2\right] + \left[(C_i)_P^2 + (C_i)_{P,\perp}^2\right] \\ &\quad\quad - 2[I_P \times (C_i)_P + I_{P,\perp} \times (C_i)_{P,\perp}] \\ &\quad = [I_P - (C_i)_P]^2 + [I_{P,\perp} - (C_i)_{P,\perp}]^2 \stackrel{\text{def}}{=} d_*^2(I, C_i). \end{aligned} \tag{4}$$

In (4), an estimation $d_*^2(I, C_i)$ is introduced for $d^2(I, C_i)$. Obviously, once online computation for I_P and $I_{P,\perp}$ that must use square root operation is required. Then $d_*^2(I, C_i)$ needs one addition, two subtractions and two multiplications for one rejection check. A larger estimated value of $d_*^2(I, C_i)$ is better. If $d_*^2(I, C_i) \geq d_{\min}^2$ holds, then C_i can be rejected safely. Therefore, how to optimally select P becomes very important. According to the tri-hedral theorem, $\theta = |\theta_1 - \theta_2|$ holds if and only if the projection axis P lies in the plane determined by I and C_i. Then, $d_*^2(I, C_i) = d^2(I, C_i)$ becomes true. This P must be the optimal projection axis for C_i because it guarantees a maximum evaluated value of $d_*^2(I, C_i)$ as $d^2(I, C_i)$ but P is not sole.

In the second way to directly project D_i onto an arbitrary P, although D_i and P are two noncoplanar straight lines as defined in solid geometry, the concept of inner product and a projection operation is still applicable. Therefore, it is clear that $(\langle D_i, P\rangle)^2 \leq \|D_i\|^2\|P\|^2$ and $(D_i)_P = \langle D_i, P\rangle/\|P\|$ holds. When P parallels to D_i, the equality holds. However, because D_i and P are not intersected, the orthogonal component $(D_i)_{P,\perp}$ cannot be used because it is not a sole value. Therefore, we have

$$\begin{aligned} d^2(I, C_i) &= \|D_i\|^2 \geq (\langle D_i, P\rangle)^2/\|P\|^2 = [(D_i)_P]^2 \\ &= [(I - C_i)_P]^2 = [I_P - (C_i)_P]^2 \stackrel{\text{def}}{=} d_\#^2(I, C_i). \end{aligned} \tag{5}$$

In (5), another estimation $d_\#^2(I, C_i)$ is introduced for $d^2(I, C_i)$. Obviously, just once online computation for I_P but $I_{P,\perp}$ is required. Then $d_\#^2(I, C_i)$ needs one subtraction, one multiplication for one rejection check. A larger estimated value of $d_\#^2(I, C_i)$ is better. If $d_\#^2(I, C_i) \geq d_{\min}^2$ holds, then C_i can be rejected safely as well. Based on the analysis above, the optimal P must be the one that parallels to D_i and it guarantees a maximum estimated value of $d_\#^2(I, C_i)$ as $d^2(I, C_i)$.

Comparing (4) with (5), it is obvious that (4) will become (5) if the term $[I_{P,\perp} - (C_i)_{P,\perp}]^2$ is discarded or (4) is more general. In addition, both (4) and (5) can approach the $d^2(I, C_i)$ as an upper bound, which implies that $d_*^2(I, C_i)$ and $d_\#^2(I, C_i)$ potentially are the same powerful as $d^2(I, C_i)$ for rejection. But (4) and (5) have very different physical meaning and they result in a different criterion for optimally selecting the projection axis P.

Then, a search flow is suggested as follows: (1) Set up an appropriate projection axis P. (2) Compute $(C_i)_P$ off-line. Rearrange the codebook along the $(C_i)_P$ sorted in ascending order. If using (4) check, compute the corresponding $(C_i)_{P,\perp}$ off-line as well. (3) For I, compute its I_P and $I_{P,\perp}$ (if necessary) once online. Find the code vector C_N that is closest to I in terms of a minimum $|I_P - (C_i)_P|$ difference and compute the "so far" $d_{\min}^2 = d^2(I, C_N)$. (4.1) Continue the search up and down around C_N. If $d_\#^2(I, C_i) \geq d_{\min}^2$, then terminate the search for the remaining upper part of codebook when $i < N$ or for the remaining lower part of codebook when $i > N$. If the search in both upper and lower directions has been terminated, search is complete. The current best-matched code vector must be the winner. Then go to Step (3) for encoding another new I. (4.2) Else, if and only if using (4), check whether $d_*^2(I, C_i) \geq d_{\min}^2$ holds or not. If this inequality holds, then reject current C_i safely. (5) When both $d_\#^2(u, v)$ and $d_*^2(u, v)$ (if used) checks fail, compute squared Euclidean distance $d^2(I, C_i)$. If $d^2(I, C_i) \geq d_{\min}^2$ holds, then reject C_i. Otherwise, update the winner index and $d_{\min}^2$ accordingly. Then go to Step (4.1) for checking another new code vector.

III. Related Previous Works

In this letter, (4) together with (5) are proposed as a unified projection method for VQ search. Some of previous works can be unified by them as explained below.

CASE 1: In previous work [3], (3) is straightforwardly used for estimating the real $d^2(I, C_i)$ in order to make a rejection. [3] needs four memory units for storing $\|C_i\|^2, \|C_i\|, \cos(\theta_2), \sin(\theta_2)$ and needs four multiplication $(\times)$ and three addition $(\pm)$ operations online for each code vector. Besides, $\cos(\theta_1)$ and $\sin(\theta_1)$ have to be computed once online as well for I, which implies once rather heavy square root operation. [3] is still more complex. On the other hand, (4) is exactly equivalent to (3) but (4) needs only two memory units for $(C_i)_P$ and $(C_i)_{P,\perp}$ and two multiplication $(\times)$ and three addition ($\pm$) operations online for each code vector. Similarly, once I_P and $I_{P,\perp}$ computation online is necessary for I. It concludes that (4) reaches the exact same results as [3] but it is more efficient than [3].

CASE 2: In previous work [4], the concept of a central axis in R^k space is introduced as: if any point $P=(P_1,P_2,\ldots,P_k)$ on a line L satisfies $P_1=P_2=\cdots=P_k$, then L is called a central axis of R^k space. For a vector $w=(w_1,w_2,\ldots,w_k)$, under the constraint of the central axis, its projection onto L will be the point $L_w=(\mu_w,\mu_w,\ldots,\mu_w)$, where μ_w is the mean of w. Then, three points w, L_w and the origin $O=(0,0,\ldots,0)$ constitute a triangle in R^k space. Clearly, the projection of w onto L is $d(L_w,O)=\sqrt{k}\times\mu_w$ and its orthogonal component is, $d(w,L_w)=\sqrt{\sum_{j=1}^{k}(w_j-\mu_w)^2}\stackrel{\text{def}}{=}V_w$ where V_w is the variance of w. Then a powerful inequality is developed by [4] as $d^2(I,C_i)\geq k(\mu_I-\mu_{C_i})^2+(V_I-V_{C_i})^2$ for rejection. On the other hand, if $P=(1,1,\ldots,1)$ is selected in (4), we have $I_P=\langle I,P\rangle/\|P\|=\sqrt{k}\times\mu_I$. Obviously, $I_{P,\perp}=\sqrt{\|I\|^2-I_P^2}=V_I$ holds. For C_i, it is the same. Then (4) will become $d^2(I,C_i)\geq(\sqrt{k}\times\mu_I-\sqrt{k}\times\mu_{C_i})^2+(V_I-V_{C_i})^2$. It concludes that (4) is a general version of [4].

CASE 3: In previous work [5], the concept of a generalized central axis in R^k space is introduced as: if any point $P=(P_1,P_2,\ldots,P_k)$ on a line L satisfies the condition $|P_1|=|P_2|=\cdots=|P_k|$, then L is defined as a generalized central axis in R^k. Under the constraint of a generalized central axis, let L_I be the projection point of I onto L, [5] concluded that $L_I=P\cdot(p_I,p_I,\ldots,p_I)$, where $p_I=\langle P,I\rangle/k$. Note: p_I is not a conventional projection here. For C_i, it is the same. Then a powerful inequality is proven in [5] as $d^2(I,C_i)\geq k(p_I-p_{C_i})^2$ for rejection. Because $k(p_I-p_{C_i})^2$ is very light computationally, it becomes practical to use multiple projection axes so as to achieve a successful rejection finally as proposed by [5], in which three projection axes have been used. On the other hand, if $P=(\pm1,\pm1,\ldots,\pm1)$ is selected in (5), we have $I_P=\langle I,P\rangle/\|P\|=\langle I,P\rangle/\sqrt{k}$. For C_i, it is the same. Then (5) will become $d^2(I,C_i)\geq[\langle I,P\rangle/\sqrt{k}-\langle C_i,P\rangle/\sqrt{k}]^2=k(p_I-p_{C_i})^2$. It concludes that (5) is a general version of [5]. Of course, (4) is also a general version of [5].

CASE 4: In previous work [6], the concept of an orthogonal transform such as Karhunen-Loeve transform or Walsh-Hadamard transform are introduced in order to exploit the energy compaction property of the orthogonal transform. Suppose a set of orthogonal basis vectors have been found by using the available codebook information as $Q=[q_1^T;q_2^T;\ldots;q_k^T]$, which has already been rearranged according to the sorted eigen values λ_j for $j=1\sim k$ in descending order. Because Q is an orthogonal basis vector set, it must satisfy that $<q_i^T,q_j^T>=0$ for $i\neq j$ and $<q_i^T,q_i^T>=1$ or $\|q_i^T\|=1$ are true. Thus, the first eigenvector q_1^T corresponding to the largest λ_1 is the first principal axis in a space spanned by Q. Then, [6] used the first principal component $z_1=q_1^TI$ and $w_1=q_1^TC_i$ and their orthogonal components for measuring the difference between I and C_i. In this case, z_1 is the projection of I to q_1^T and the coordinate of its projection point is z_1q_1. Therefore, the distance between I and its projection point is $\mathrm{V}_I=d(I,z_1q_1)$. For C_i it is the same. Then a powerful inequality is proposed by [6] as $d^2(I,C_i)\geq(z_1-w_1)^2+(V_I-V_{C_i})^2$. On the other hand, if the first eigen vector q_1^T is just viewed as a mathematical projection axis while not taking its physical meaning into account, we have $P=q_1^T$. It is clear that $I_P=\langle I,P\rangle/\|P\|=\langle I,q_1^T\rangle/\|q_1^T\|=\langle I,q_1^T\rangle=z_1$, $I_{P,\perp}=\sqrt{\|I\|^2-I_P^2}=\sqrt{\|I\|^2-z_1^2}=\sqrt{\sum_{j=1}^{k}I_j^2-2z_1(q_1^TI)+z_1^2\|q_1^T\|^2}=\sqrt{\sum_{j=1}^{k}(I_j-z_1q_{1,j})^2}=d(I,z_1q_1)=V_I$. For C_i, it is the same. Then (4) will become $d^2(I,C_i)\geq(z_1-w_1)^2+(V_I-V_{C_i})^2$. It concludes that (4) is a general version of [6]. But the deduction method in this letter is much simpler than [6].

CASE 5: In previous work [7], L_2 norm difference is used to make a rejection. For I, its L_2 norm is $\|I\|=\sqrt{\sum_{j=1}^{k}I_j^2}$. For C_i it is the same. Then [7] proposed a powerful inequality (Algorithm 2) as $d(I,C_i)\geq|\|I\|-\|C_i\||$ for rejection. On the other hand, if $P=(0,0,\ldots,0)$ in a broad sense is selected in (4), we have $I_P=0$ and $I_{P,\perp}=\sqrt{\|I\|^2-I_P^2}=\sqrt{\|I\|^2-0}=\|I\|$. For C_i, it is the same. Thus, (4) will become $d^2(I,C_i)\geq0+[I_{P,\perp}-(C_i)_{P,\perp}]^2=(\|I\|-\|C_i\|)^2$. It concludes that (4) is a general version of [7].

CASE 6: In previous work [8], a temporary difference is defined as $d_1^2(I,C_i)=d^2(I,C_i)-\|I\|^2$. Then a powerful inequality is given as $d_1^2(I,C_i)\geq\|C_i\|^2-2\|I\|\|C_i\|=\|C_i\|(\|C_i\|-2\|I\|)$. If $\|C_i\|(\|C_i\|-2\|I\|)\geq d_{1\min}^2$ holds, C_i can be rejected safely, where $d_{1\min}^2$ instead of $d_{\min}^2$ is used. On the other hand, if $P=(0,0,\ldots,0)$ is also selected in (4) and $\|I\|^2$ is subtracted from both sides of it $d^2(I,C_i)-\|I\|^2\geq\|C_i\|^2-2\|I\|\|C_i\|$, holds as well. It concludes that (4) is a general version of [8].

Case 7: : In previous work [9], the famous partial distortion search (PDS) method proposed a powerful inequality as $d^2(I,C_j)\geq\sum_{j=1}^{m}(I_j-C_{i,j})^2$ for rejection, where $m\leq k$. On the other hand, without losing the generality, if $P=(1_{(1)},\ldots,1_{(m)},0_{(m+1)},\ldots,0_{(k)})$ is selected in (5), we have $I_P=\langle I,P\rangle/\|P\|=\sum_{j=1}^{m}I_j/\sqrt{m}$. For C_i, it is the same. Then (5) will become $d^2(I,C_i)\geq[\sum_{j=1}^{m}(I_j-C_{i,j})]^2/m$. According to the Cauchy-Schwarz inequality, it is obvious that $(x_1+x_2+\cdots+x_k)^2\leq k\times(x_1^2+x_2^2+\cdots+x_k^2)$ is true. Thus $[\sum_{j=1}^{m}(I_j-C_{i,j})]^2/m\leq\sum_{j=1}^{m}(I_j-C_{i,j})^2$ is true. This implies that the upper limit of rejection power of (5) can approach the rejection power of PDS but (5) is much lighter computationally.

IV. Experimental Results

Simulations with MATLAB are executed for four typical 8-bit, 512×512 standard images (Lena, F-16, Pepper, and Baboon). Image block is 4×4 and its elements are in a raster order. Codebooks are generated using Lena image as a training set based on the Kohonen's self-organizing map (SOM) method used in [10]. Because how to select an optimal projection axis for each code vector online is out of the scope of this letter, the central axis P1 $=(1,1,\ldots,1)$ as proposed in [4] is used for all code vectors as well for simplicity. The reason is that the

TABLE I
RATIO OF THE REDUCED SEARCH SPACE AFTER EACH CHECK STEP COMPARED TO FS (100%)

Size	Lena				F-16			
	Using Eq.4		Using Eq.5		Using Eq.4		Using Eq.5	
	S1	S2	S1	S2	S1	S2	S1	S2
128	9.18	1.88	9.18	3.30	8.13	1.61	8.13	3.51
256	7.52	1.56	7.52	2.87	6.70	1.37	6.70	3.24
512	6.40	1.27	6.40	2.54	5.93	1.15	5.93	3.13
1024	5.37	0.96	5.37	1.87	5.42	1.03	5.42	2.76
Size	Pepper				Baboon			
	Using Eq.4		Using Eq.5		Using Eq.4		Using Eq.5	
	S1	S2	S1	S2	S1	S2	S1	S2
128	9.94	1.82	9.94	3.97	22.2	6.25	22.2	14.6
256	8.42	1.57	8.42	3.70	20.5	5.91	20.5	14.5
512	7.58	1.36	7.58	3.60	19.7	5.46	19.7	14.7
1024	7.01	1.17	7.01	3.09	18.8	5.29	18.8	13.6

projections of a vector onto $P1$ have a clear physical meaning as the mean and the variance. In fact, $P1$ exploits the potential power of using the statistical averaging, which is a well-known and very effective method. When using (4) check for a possible rejection, two steps are executed sequentially, which are the step1 (S1) as $[I_{P1} - (C_i)_{P1}]^2 \geq d^2_{\min}$ and the step2 (S2) as $[I_{P1} - (C_i)_{P1}]^2 + [I_{P1,\perp} - (C_i)_{P1,\perp}]^2 \geq d^2_{\min}$. If S1 check is successful, S2 check can be skipped to save more computational burden. However, there exist two problems for the S2 check by using (4), which are the computation in real form and once square-root operation online for $I_{P,\perp}$. For comparison, (5) check can be used twice for a possible rejection with the same memory requirement. Then, another projection axis can be selected as $P2 = (1, 1, -1, -1, \ldots, 1, 1, -1, -1)$, which is a generalized central axis as proposed in [5] and also exploits the potential power of the statistical averaging in a different direction. $P1$ and $P2$ are orthogonal. Similarly, two steps are also executed sequentially as S1: $[I_{P1} - (C_i)_{P1}]^2 \geq d^2_{\min}$ and S2: $[I_{P2} - (C_i)_{P2}]^2 \geq d^2_{\min}$. Obviously, the S1 check in both ways is the exact same but S2 check by (5) is lighter. The experimental results are summarized in Table I, where the search efficiency in the form of a ratio is evaluated by how small the search space can be reduced or how many code vectors are left after each check step completed with FS (100%) as a relative baseline rather than encoding time to exclude the effect of programming skills. A smaller ratio is better.

From Table I, it can be seen that the check using (4) is much more search-efficient than the check using (5), which comes from using a different S2 check. A reasonable explanation could be that $[I_{P1} - (C_i)_{P1}]^2 + [I_{P1,\perp} - (C_i)_{P1,\perp}]^2$ is more powerful than $[I_{P1} - (C_i)_{P1}]^2$ in (4) while $[I_{P1} - (C_i)_{P1}]^2 \geq d^2_{\min}$ is almost the same powerful as $[I_{P2} - (C_i)_{P2}]^2 \geq d^2_{\min}$ in (5) due to I and C_i are random. But the square root operation for online computing $I_{P,\perp}$ in (4) is a potential problem.

Based on the analysis in Section II, it is clear that the central axis, the generalized central axis or the first principal axis is generally not an optimal projection axis. And a fixed projection axis for all code vectors is obviously too coarse. However, it is impractical to select an optimal projection axis for each code vector by a lot of online processing. To make a tradeoff, a promising consideration could be to appropriately divide all code vectors into several small groups and then to select a suboptimal projection axis for each group by a little online cost to seek for a higher search efficiency.

V. CONCLUSION

In this letter, two contributions are made. First, a unified projection method is proven mathematically, which can be viewed as a general version of several types of previous works that used the projection concept by suitably selecting a projection axis P. As a result, the constraints such as the concept of a central axis for selecting a projection axis can be completely removed. Any projection axis P is possible to be adopted. Second, two criteria on how to select an optimal P for a promising rejection have also been given as (1) P parallels to the difference vector $D_i = I - C_i$ or (2) P lies in the plane determined by I and C_i. These two criteria are most important because they mathematically show that the upper bound of Euclidean distance estimation that is possible to be achieved by using a vector projection concept is in fact the real Euclidean distance itself. In order to obtain a larger estimated value of Euclidean distance for realizing a rejection easier, it is clear that the "if and only if" way is to let P approach either of the two criteria described above as close as possible. Comparing (5) with (4), the former selecting way is more advantageous for it can avoid computing the difference between orthogonal components. For practical applications of VQ, the remaining problem to be resolved is how to select an optimal or suboptimal P by a little online processing.

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

[3] J. Mielikainen, "A novel full-search vector quantization algorithm based on the law of cosines," *IEEE Signal Processing Lett.*, vol. 9, pp. 175–176, June 2002.

[4] S. J. Baek, B. K. Jeon, and K. M. Sung, "A fast encoding algorithm for vector quantization," *IEEE Signal Processing Lett.*, vol. 4, pp. 325–327, Dec. 1997.

[5] S. J. Baek, M. J. Bae, and K. M. Sung, "A fast vector quantization encoding algorithm using multiple projection axes," *Signal Process.*, pp. 89–92, 1999.

[6] S. J. Baek and K. M. Sung, "Two fast nearest neighbor searching algorithms for vector quantization," *IEICE Trans. Fund.*, vol. E84-A, pp. 2569–2575, 2001.

[7] C. M. Huang, Q. Bi, G. S. Stiles, and R. W. Harris, "Fast full search equivalent encoding algorithm for image compression using vector quantization," *IEEE Trans. Image Processing*, vol. 1, pp. 413–416, July 1992.

[8] K. S. Wu and J. C. Lin, "Fast VQ encoding by an efficient kick-out condition," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, pp. 59–62, Feb. 2000.

[9] 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, 1985.

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