Available online at www.sciencedirect.com
ScienceDirect
Signal Processing 87 (2007) 799–810
SIGNAL PROCESSING
www.elsevier.com/locate/sigpro

A novel approach for vector quantization using a neural network, mean shift, and principal component analysis-based seed re-initialization

Chin-Chuan Han[a,*], Ying-Nong Chen[b], Chih-Chung Lo[c], Cheng-Tzu Wang[d]

[a]Department of Computer Science and Information Engineering, National United University, Miaoli, Taiwan
[b]Department of Computer Science and Information Engineering, National Central University, Taoyuan, Taiwan
[c]Department of Informatics, Fo Guang College of Humanities and Social Sciences, Ilan, Taiwan
[d]Department of Computer Science, National Taipei University of Education, Taipei, Taiwan

Received 6 December 2005; received in revised form 15 July 2006; accepted 7 August 2006
Available online 7 September 2006

Abstract

In this paper, a hybrid approach for vector quantization (VQ) is proposed for obtaining the better codebook. It is modified and improved based on the centroid neural network adaptive resonance theory (CNN-ART) and the enhanced Linde–Buzo–Gray (LBG) approaches to obtain the optimal solution. Three modules, a neural net (NN)-based clustering, a mean shift (MS)-based refinement, and a principal component analysis (PCA)-based seed re-initialization, are repeatedly utilized in this study. Basically, the seed re-initialization module generates a new initial codebook to replace the low-utilized codewords during the iteration. The NN-based clustering module clusters the training vectors using a competitive learning approach. The clustered results are refined using the mean shift operation. Some experiments in image compression applications were conducted to show the effectiveness of the proposed approach.

Keywords: Vector quantization; ELBG algorithm; Neural network; Mean shift; Principal component analysis

1. Introduction

Vector quantization (VQ) is an efficient and simple approach for data compression. It was derived from Shannon's rate-distortion theory [1], and has been successfully applied in image compression and speech coding. It encodes images by scalars instead of vectors for obtaining better performance. In the procedure for image compression, VQ first partitions an image into several blocks to form a vector set (input vectors). The input vectors are individually quantized to the closest codeword from a codebook. This codebook (e.g. a set of codewords) was generated from a set of training vectors by using the clustering techniques. An image was encoded by the indices of codewords and decoded by a table-look-up technique.

In general, VQ partitions a vector set X of size N_p into a codebook C of size N_c. That is $X = \{x_1, x_2, \ldots, x_{N_p}\}$, $C = \{c_1, c_2, \ldots, c_{N_c}\}$, and the dimensions of all vectors are m. Here, m is the size of an encoded image block, and $N_c \ll N_p$. To quantize

*Corresponding author. Tel.: +886 3 7381258; fax: +886 3 7354326.
E-mail address: cchan@nuu.edu.tw (C.-C. Han).

0165-1684/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.sigpro.2006.08.006

each vector x_k, a codeword c_j is selected with the shortest distance $d(x_k, c_j)$ between x_k and c_j. The Euclidean distance is the most used metric, i.e., $c_j = q(x_k) = \arg\min_{c_i \in C} \|x_k - c_i\|$. Assume a universal set U of all possible codebooks. The better codebook is obtained by minimizing the square of errors between two sets X and C,

$$C = \arg\min_{q_j \in U} \left(\frac{1}{N_p} \sum_{k=1}^{N_p} d(x_k, q_j(x_k))^2 \right). \quad (1)$$

Linde et al. [1] proposed the famous Linde–Buzo–Gray (LBG) algorithm to find the codebook for image compression. However, the results of VQ methods are much affected by the initialization of the codebook. They are frequently trapped in the local solution resulting in poor performance. Besides, the low-utilized codewords in the generated codebook distort the decoded images. There three problems are still present in the LBG algorithm. Patane and Russo [2] proposed the enhanced LBG (ELBG) algorithm to find an optimal codebook by means of shifting the low-utilized codewords to another ones with high utility. This is a split-and-merge-based algorithm for improving the utility of codewords. Kaukoranta et al. [3] also proposed an iterative algorithm combining with split and merge operations (ISM) for the generation of a codebook. Haber and Seidel [4] modified the LBG algorithm, called ILBG, to reduce the codebook errors by a few additional iteration steps. Huang et al. [5] proposed a novel algorithm to improve the codeword utility and the training performance by combining the genetic algorithm and the simulated annealing technique. The crossover and mutation operations were based on the simulated annealing process. The local optimal solution was avoided.

Recently, many researchers have tried to solve the problems in finding the optimal solution using the neural network-based learning approaches. Lin and Yu [6] proposed a centroid neural network adaptive resonance theory (CNN-ART) to improve the performance of VQ. CNN-ART, an unsupervised and competitive neural network model, considered the weights of neurons as the codewords. Although CNN-ART relieved the dependence on the initialization of the codebook, it was still affected as can be seen from the results. Besides, there is a low-utility problem in their proposed approach with poor initialization. Laha et al. [7] designed a codebook using a self-organizing feature map technique. During the training process, the weights of nodes were considered to be the codewords. All weights of nodes built up the codebook. However, the block-effects frequently distorted the reconstructed images. This problem was solved by a polynomial surface fitting technique [7].

On the other hand, researchers tried to speed up the process of finding the optimal solution. The tree search vector quantizer (TSVQ) was the improbable algorithm using a tree structure [8]. Chang and Lin [9] eliminated most of the impossible codewords to decrease the matched number for speeding up the searching process. Chan and Ma [10] proposed a fast maximum descent (MD) approach to quickly find the codebook. Pan et al. [11] modified the L2-norm pyramid encoding approach to decrease the searching time. Chen [12] utilized the fuzzy reasoning technique to predict the codewords for improving the searching performance. Huang and Chang [13] proposed a color finite-state LBG (CFSLBG) algorithm to reduce the searching time of the LBG algorithm.

Since the compression performance is much affected by the codebook, the codebook generation has a key role in VQ. In summary, four problems frequently occur in the clustering process. They are (1) the *initial condition*, (2) the *local optimum solution*, (3) the *low-utilized codeword*, and (4) the *sequence of training vectors* problems. Given a training set $X = \{x_1, x_2, \ldots, x_{N_p}\}$, N_p elements are partitioned into N_c subsets and the sum of squared errors in Eq. (1) should be minimized. There are $(N_c)^{N_p}/N_c$ ways for partition. It is infeasible to find the optimal partition by an exhaustive searching approach. Iterative optimization is the most frequently used approach. N_c initial codewords $c_1, c_2, \ldots, c_{N_c}$ are randomly guessed. At each iteration, N_p training vectors are sequentially assigned to the nearest codeword and a new codeword is recalculated as the mean of all instances belonging to the same cluster. These two steps are repeated until the codewords stabilize. The iterative procedure guarantees local but not global optimization. The found solution highly depends on the initial condition C^i and the sequence of training vectors $x_1, x_2, \ldots, x_{N_p}$. They are the well-known problems of initial condition, local optimization, and sample sequence. In addition to the sum-of-squared-error criterion, the codeword utility is another indication to evaluate the total distortions related to codeword c_j. The equalization of the codeword distortions is equivalent to the equalization of the codeword

utilities. According to the consequences in Ref. [2], the aim of the clustering operations is to obtain the equal distortion for each codeword in optimal VQ. Therefore, discarding the low-utility codewords and splitting the high-utility codewords into smaller ones are the effective strategies to solve the low-utility problem.

In this paper, a hybrid algorithm is proposed to find a better codebook for image compression applications. Three modules, a neural net (NN)-based clustering, a mean shift (MS)-based refinement, and a principal component analysis (PCA)-based seed re-initialization, were integrated in this study. It is abbreviated as PCA-NN-Meanshift (PNM). Initially, a codebook of size N_c was randomly generated. These codewords were inputted into the weights of a NN. Next, the training vectors were clustered and the weights were adapted up to a converged state. In addition, a MS-based operation was performed to refine the codeword in each cluster. However, a local optimal solution was frequently obtained. Besides, codewords with low utility were frequently generated. A PCA-based process was performed to analyze the sample distribution. New seeds were generated to replace the codewords with low utility. These three modules were repeatedly executed to find the better solution. The rest of this paper is organized as follows: The background of ELBG and CNN-ART are briefly reviewed in Section 2. In Section 3, the modules, NN-based clustering, MS-based refinement, and PCA-based seed re-initialization, were designed to find the better codebook. In Section 4, some experiments in the image compression application were conducted to show the feasibility of proposed method. Some conclusions are given in Section 5.

2. Background

In this paper, the ELBG and CNN-ART algorithms were improved to obtain a better codebook for VQ. Following is a brief description of the backgrounds of these two algorithms.

2.1. Enhanced LBG algorithm

An ELBG algorithm was proposed by Patane and Russo to improve the LBG method [2]. They applied a utility rate for each codeword to enhance the performance of the LBG algorithm. The main improvement was based on a *shifting of codewords attempts* (SoCA) procedure. Basically, a codeword c_l with a low-utility rate was heuristically shifted to a nearby codeword c_h with a high rate. Suppose that a training set X assigned to codeword c_h was bounded in a hyper-box $I=[x_{1l},x_{1u}]\times[x_{2l},x_{2u}]\times\cdots\times[x_{ml},x_{mu}]$, and codeword c_h was the center vector of set X. Codewords c_l and c_h were recomputed as $c_l=[x_{1l}+\frac{1}{4}(x_{1u}-x_{1l}),x_{2l}+\frac{1}{4}(x_{2u}-x_{2l}),\ldots,x_{ml}+\frac{1}{4}(x_{mu}-x_{ml})]$ and $c_h=[x_{1u}-\frac{1}{4}(x_{1u}-x_{1l}),x_{2u}-\frac{1}{4}(x_{2u}-x_{2l}),\ldots,x_{mu}-\frac{1}{4}(x_{mu}-x_{ml})]$. All vectors in set X were re-clustered to these two new codewords. That means two codewords split the training vectors in set X. This is a split-and-merge process. The largest cluster was split into two clusters by two new codewords c_l and c_h, and the smallest cluster was merged to another cluster. The SoCA procedure is summarized as follows:

1. *Check the stopping criterion*: Check if at least one codeword with a utility rate of less than 1 has been shifted in the previous iteration. If not;terminate the procedure.
2. *Codeword selection*: Select two codewords, a codeword c_h with a utility rate greater than 1 and a codeword c_l with a rate less than 1, to execute the codeword shifting process.
3. *Codeword shifting and local rearrangement*: Shift codeword c_l to codeword c_h. After that, adjust these two codewords by the traditional LBG algorithm under a termination criterion with a higher threshold.
4. *Quantization error estimation*: Calculate the expected values of quantization errors (QE) before and after shifting. If value QE has decreased after shifting, confirm the shifting process. Otherwise, reset codewords c_l and c_h to their original positions.

Repeat the above steps until the stop criterion is satisfied. This ELBG algorithm is used to solve the problem of local optimal solution.

2.2. Centroid neural network adaptive resonance theory (CNN-ART)

Lin and Yu [6] proposed a novel algorithm, CNN-ART, to generate a codebook. It is an unsupervised competitive learning algorithm, and is an extension of Grossberg's adaptive resonance theory [14]. Basically, the CNN-ART algorithm considered the synaptic weight vectors in each

neuron as codewords. The gradient-descent-based algorithm was used as the weight updating rule. CNN-ART started at a single node, i.e. the size of codebook was initialized to one. The winner node was rewarded with a positive learning gain and the loser nodes were punished with a negative one for the competitive learning rules. This approach is a *threshold criterion*-based clustering approach. A centroid node is increased as a new cluster when the Euclidean distances between an input vector and the existing nodes are larger than a vigilance value. The CNN-ART approach repeated the incremental process until the codebook size was N_c. The algorithm is summarized as follows:

1. *Initialization*: Initialize the following variables: the codebook size N_c, the initial codebook $C^0 = \emptyset$, the training set $X = \{x_k : k = 1, 2, \ldots, N_p\}$, a pre-defined threshold v, and the iteration index $t = 0$.
2. *Clustering*: Given the codebook C^t at iteration t, calculate the Euclidean distances between vector x_k and the weights in the network. Assign vector x_k to the node with the smallest distance.
3. *Node increment or weight updating*: If the smallest distance was larger than threshold v, and the node number was smaller than value N_c, generate a new node. Otherwise, update the weights of a winner with the rewarding rules and those of losers with the punishing rules.
4. *Check the stop criterion*: When the state is stable, the process is terminated.
5. *Generate the codebook*: If the codebook size was N_c, and a converged and stable state occurred, the codebook is assigned as the set of all nodes' synaptic weights.

Due to the scenarios of the ELBG algorithm and the CNN-ART algorithm, four problems should be solved in the VQ process. To overcome these problems and to improve the performance of the ELBG or the CNN-ART algorithms, a hybrid algorithm was developed in the following.

3. PNM for VQ

The architecture of PNM is composed of a PCA-based seed re-initialization module, an NN-based clustering module, and an MS-based refinement module. They were iteratively performed to find the optimal solution.

3.1. NN-based clustering

A simple MINNET network is a competitive learning algorithm. It determines the nearest distance between an input vector and those in the neuron's weights for the output layer. The dimensions of the input layer is m the same as the number of each neuron's synaptic weights because of the full connection between them. Therefore, the weights in a neuron are considered as a codeword. The main difference between the PNM and the CNN-ART architecture is the number of initial neurons. CNN-ART repeatedly increases the neurons until N_c nodes (the codebook size). Whereas N_c neurons are initialized in PNM, and their weights are randomly initialized. The functions of MINNET in PNM are the same as those in CNN-ART. All neurons are completely interconnected in MINNET. Each neuron got the values from its original neuron and the lateral inhibition $(-\varepsilon)$ from the other neurons. The output value $O_j^{(t)}$ of the jth neuron at iteration t is thus given as follows:

$$O_j^{(t)} = f_t\left(O_j^{(t-1)} - \varepsilon \sum_{i \neq j} O_i^{(t-1)}\right) \quad \text{and}$$
$$i, j = 1, 2, \ldots, N_c, \quad t \geqslant 1, \tag{2}$$

$$f_t(\beta) = \begin{cases} \beta & \text{if } \beta < 0, \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Here, the initial condition was set as

$$O_j^{(0)} = \|x_k - w_j\|. \tag{4}$$

Value N_c denotes the number of neurons in net MINNET, and $\varepsilon > 1/N_c$. Each node was repeatedly compared with the other nodes until only a negative output was generated. Meanwhile, the other neurons all outputted zero. The node with the negative output was the desired codeword.

Next, let us describe the learning rules in PNM. Similar to the rules in CNN-ART and LBG algorithms, the rewarding equation is written below:

$$w_j^{(t)} = w_j^{(t-1)} + \frac{1}{|c_j| + 1}[x_k - w_j^{(t-1)}],$$
$$k = 1, 2, \ldots, N_p, \quad j = 1, 2, \ldots, N_c, \tag{5}$$

and value $1/(|c_j| + 1)$ is its learning rate.

The above learning approach plays a clustering role in PNM. The training vectors were sequentially clustered with the cluster centers (e.g., codewords, the weights of nodes).

3.2. MS-based refinement

After the NN-based clustering procedure, the MS operation [15] was performed on each cluster to refine the codeword. In this operation, the multivariate kernel density estimate with an observation window W was estimated as

$$\hat{f}(\mathbf{x}) = \frac{1}{N_w h^m} \sum_{k=1}^{N_w} k_e\left(\frac{\mathbf{x} - x_k}{h}\right). \quad (6)$$

Here, symbol $\mathbf{x}$ is an observation sample, values m, N_w, and h represent the window's dimensional number, the sample number in window W, and the window's radius, respectively. The Epanechnikov kernel function was applied in this procedure as

$$k_e(\mathbf{x}) = \begin{cases} \frac{1}{2} c_m^{-1}(m+2)(1-\mathbf{x}^T\mathbf{x}) & \text{if } \mathbf{x}^T\mathbf{x} < 1, \\ 0 & \text{otherwise.} \end{cases} \quad (7)$$

c_m is the volume of unit sphere. The codeword shifted towards to the next position by the density gradient of Eq. (6) as

$$M_h(\mathbf{x}) = \frac{h^2}{m+2} \frac{\hat{\nabla} f(\mathbf{x})}{\hat{f}(\mathbf{x})}, \quad (8)$$

$\hat{\nabla} f(\mathbf{x})$ is a density gradient estimate. According to the conclusions in Ref. [15], the sample MS in a uniform kernel centered on $\mathbf{x}$ is the normalized gradient of a kernel density estimate. Consider a region $S_h(\mathbf{x})$ is a hypersphere of radius h in which point $\mathbf{x}$ is a center of N_w samples. The sample MS is defined as

$$M_h(\mathbf{x}) = \frac{1}{N_w} \sum_{x_k \in S_h(\mathbf{x})} (x_k - \mathbf{x}). \quad (9)$$

Value h was set as half of the second eigenvalue λ_2 in this study. Compute the MS vector $M_h(\mathbf{x})$ using the training samples belonging to the codeword c_j, and shift to the next position. Repeat the moving process to refine the codeword.

3.3. PCA-based seed re-initialization

Since the results of CNN-ART algorithm are influenced by the initialization problem, a local optimal solution is obtained. Besides, the codewords in low utility are frequently generated from the local optimal solution. As mentioned in the previous section, there are four problems that frequently occur in many proposed approaches. In order to remedy these problems, a PCA-based seed re-initialization procedure was performed to generate a new codebook. Using this procedure, the inference of initial conditions was reduced. In addition, the probability of finding an optimal solution was increased, and the number of low-utilized codeword was thus decreased. The clustering results of NN-based and MS-based procedures were applied in the seed re-initialization module. Generally, the large clusters were split into several smaller clusters, and the small clusters with low utility were discarded and merged into a bigger cluster. The splitting rules are designed as follows: consider N_s split clusters and N_m merged clusters. N_m cluster centers were discarded, and N_m new cluster centers were generated for N_s clusters. On average, N_m/N_s new cluster centers were generated for each larger cluster. The clusters were sorted based on their sizes. Thereafter, the largest N_s clusters and the smallest N_m clusters were selected to split and discarded in this module, respectively. In this study, the number of new centers depended on a generation rate ρ. $N_m = \rho N_c$ new centers were generated in each iteration, and $N_s = 3$. In addition, value ρ was decreased by the distortion δ between two iterations. Next, the PCA-based strategy to generate new cluster centers is designed as follows.

3.3.1. PCA-based seed selection

PCA is the most popular technique in many applications. Suppose there are M training samples possessing the feature vectors of length m. Image blocks of size 4 by 4, e.g. $m = 16$, are frequently encoded in image compression. The covariance matrix Σ is defined as $\Sigma = \sum_{k=1}^{M} (x_k - \mu)(x_k - \mu)^T$, where μ denotes the sample center. The major K eigenvectors ϕ_i corresponding with the largest K eigenvalues λ_i comprise the major distributions of these samples. The samples can be represented by the K eigenvectors called bases with the minimal errors. That means the new centers located at the major bases have higher probabilities than the other randomly selected ones. Consider a specified cluster c_j of a larger size which was split by N_m/N_s new centers. The samples belonging to cluster c_j generated the first K major bases, $K = 3$ in this study. The samples were projected to a specified axis, ϕ_i, $i = 1, 2, 3$, to obtain the temporary point with scalar values. Those temporary points were re-clustered into N_m/N_s clusters by the traditional K-mean clustering method with 1D feature values. The clustered centers of temporary points were the

candidates of new centers. The temporary center on the first axis ϕ_1 with 1D value λ was also a point with the coordinate $\lambda\phi_1$ of dimension m in the original space. Similarly, the samples were projected and were re-clustered on the other axes to obtain the new candidate centers. The distance variances of the $3N_m/N_s$ candidate clusters were calculated and sorted. Finally, N_m/N_s candidate centers with the first N_m/N_s smallest variances were selected to be the new centers.

3.3.2. Adaptive learning rate for seed selection

In this study, the adaptive learning rules were designed to avoid the divergence. The generation rate ρ was adapted due to the distortion δ between two iterations. The distortion δ is defined as

$$\delta = \frac{(1/N_p)\sum_{k=1}^{N_p} d(x_k, q^{t-1}(x_k)) - (1/N_p)\sum_{k=1}^{N_p} d(x_k, q^{t}(x_k))}{(1/N_p)\sum_{k=1}^{N_p} d(x_k, q^{t-1}(x_k))}, \tag{10}$$

q is the mapping function for vector x_k and its corresponding codeword. The adaptive rules described in [16] were utilized to vary the learning rate. Three parameters ξ, κ, and η should be determined and three conditions were considered as follows:

1. If the distortion δ increased by more than parameter $\xi = 0.04$, the learning rate was multiplied by parameter $\eta = 1.05$. The rate ρ would be high to generate more possible new seeds. In addition, the new seeds generated in this iteration were discarded.
2. If the distortion δ decreased, the new seeds were accepted. The learning rate was multiplied by parameter $\kappa = 0.9$. The number of new seeds was decreased according to the decreasing rate.
3. If the distortion increased by less than parameter ξ, the rate was unchanged for finely adapting the seed positions.

In summary, these three modules were iteratively performed. The NN module played a clustering role, the MS module refined the codewords, and the PCA module assigned the new and better seeds. The PCA module tried to find the possible codewords near to the optimal solution. According to the sample distribution, the large clusters were split into several smaller clusters to reduce the distortion. Besides, it discarded the codewords with low utility and re-assigned the better ones with high utility. PCA module could solve the problems of codebook initialization, codeword utility, and local optimization. The MS module moved each codeword toward a better position with a high sample density. It could improve the performance of codeword utility and solution optimization. Since these modules were iteratively performed, a near global optimal solution was found, the low-utilized codewords were decreased, the poor initial seeds were discarded, and the influences of sample sequences were decreased. Thereafter, the proposed iterative scheme could solve the four problems of clustering and designed as follows:

3.3.3. Algorithm of PNM

Input: A training set $X = \{x_1, x_2, \ldots, x_{N_p}\}$ of size N_p.
Output: A set of cluster centers $C = \{c_1, c_2, \ldots, c_{N_c}\}$ of size N_c.
Step 1: Initialize the parameter ρ.
Step 2: Cluster the training samples sequentially to obtain the cluster centers using the neural net-based clustering.
Step 3: For each codeword c_j, compute the mean shift vector using the training samples in cluster c_j, and shift to the next position.
Step 4: PCA-based seed re-initialization.
- *Step* 4.1: Determine N_s split and $N_m = \rho N_c$ merged clusters based on their size, e.g. codeword utility.
- *Step* 4.2: For each split cluster c_j:
  1. Compute the eigenvectors and eigenvalues from the samples in cluster c_j.
  2. Project the samples to the first three eigenvectors ϕ_1, ϕ_2, and ϕ_3, e.g. projection axes.
  3. On each projection axis, the samples were clustered using the 1D projected values.
  4. Select N_m/N_s candidate centers with the smallest variances to be the new seeds in the next iteration.
- *Step* 4.3: Ignore N_m codewords with low utility.

Step 5: Re-calculate the parameters ρ and δ.
Step 6: Repeat Steps 2 to 6 until a converged state occurs, e.g. $\delta < 0.01$.

4. Experimental results

In this section, some experiments were conducted to show the efficiency of the proposed method. The PSNR value is the popular measurement to evaluate the compressing algorithms. The benchmark images of 512 by 512 pixels used in the experiments are shown in Fig. 1. In this study, several experiments were designed to test the performance of the algorithm compared with others.

The first three experiments were conducted to show the effectiveness of the found codebooks. First, an experiment was designed for the evaluation of the codebook size. Two images 'Lena' and 'Pepper' were encoded using codebook sizes 32, 64, 128, and 256. The codebooks were generated by the LBG [1], ELBG [2], GVQ [5], GSAVQ [5], CNN-ART [6], CNN-ART-LBG, ILBG [4], ISM [3], MD [10], and PNM approaches, respectively. The comparisons using PSNR-based measurement were made between the algorithms as shown in Table 1. In this table, the CNN-ART-LBG algorithm is an approach combining the CNN-ART and LBG approaches. Here, the codebook was first generated from the CNN-ART algorithm, and the final codebook was refined by the LBG algorithm. Similarly, the GSAVQ algorithm is an approach using the genetic algorithm encoding and the simulated annealing technique. The proposed approach is clearly superior to the others. Next, two experiments for the codebook generality were conducted. The 'Lena' image was used to be the training image and to generate a codebook in the second experiment. The codebook was used to encode and decode the images 'Pepper', 'F-16', 'Sailboat', and 'Tiffany' as shown in Fig. 1(b)–(e). The PSNR values generated by 10 algorithms are tabulated in Table 2. Similar to the second

Fig. 1. The images used in this study: (a) 'Lena', (b) 'Pepper', (c) 'F-16', (d) 'Sailboat', (e) 'Tiffany', (f) 'Goldhill', (g) 'Toys', (h) 'Zelda', (i) 'Girl', (j) 'Couple', (k) 'Baboon', and (l) 'Aerial'.

Table 1
The comparison of 10 algorithms for encoding two images 'Lena' and 'Pepper' in various codebook sizes

Size	LBG	ELBG	GVQ	GSAVQ	C-ART	CAL	ILBG	ISM	MD	PNM
(a) *Image* '*Lena*'										
32	25.77	27.85	27.72	27.90	26.47	26.57	27.51	26.35	27.56	28.86
64	27.65	28.93	28.70	28.73	28.05	28.15	28.12	27.92	28.61	30.54
128	28.74	29.80	29.84	30.00	28.94	29.04	29.02	29.01	29.65	31.66
256	30.19	32.10	32.21	32.39	30.69	30.89	31.05	30.55	32.05	32.76
(b) *Image* '*Pepper*'										
32	25.67	27.65	27.62	27.61	26.27	26.47	27.42	26.24	27.48	28.75
64	27.64	28.53	28.75	28.52	28.15	28.23	28.03	28.16	28.51	30.32
128	28.54	29.85	29.75	30.13	28.56	28.94	28.99	28.91	29.55	31.25
256	30.49	32.32	32.32	32.09	30.25	30.09	31.98	30.45	32.01	32.62

C-ART, CNN-ART; CAL, CNN-ART-LBG.

Table 2
The PSNR values for various images encoded by the codebook generated by the training vectors of image 'Lena'

Size	LBG	ELBG	GVQ	GSAVQ	C-ART	CAL	ILBG	ISM	MD	PNM
(a) *Image* '*Pepper*'										
32	24.62	26.88	26.69	26.73	25.17	25.37	25.84	25.25	26.55	27.97
64	26.48	27.95	27.61	27.72	27.02	27.15	27.12	26.95	27.48	28.94
128	27.60	28.73	28.74	28.86	27.84	27.96	28.01	27.85	28.65	29.86
256	29.11	31.09	31.36	31.44	29.89	30.01	30.84	29.85	31.03	31.91
(b) *Image* '*F*-16'										
32	25.12	27.85	27.29	27.53	25.52	25.92	26.85	25.64	27.15	28.01
64	26.95	28.99	28.01	28.52	27.35	27.75	27.85	27.41	27.94	29.55
128	28.01	29.83	29.14	29.56	28.51	28.81	28.86	28.55	29.06	30.95
256	29.91	31.92	31.16	31.54	30.31	30.71	30.78	30.35	31.05	32.81
(c) *Image* '*Sailboat*'										
32	23.02	25.75	25.19	25.33	23.42	23.82	24.85	23.65	25.15	26.03
64	24.75	26.79	26.02	26.32	25.05	25.55	25.74	25.44	26.01	27.65
128	25.81	27.63	27.04	27.46	26.21	26.62	26.95	26.43	27.02	28.85
256	27.71	29.82	29.06	29.44	28.11	28.55	28.74	27.43	29.03	30.75
(d) *Image* '*Tiffany*'										
32	25.72	27.78	27.65	27.73	26.02	26.42	27.13	26.34	27.52	27.78
64	27.58	28.85	28.71	28.69	27.85	28.15	28.43	27.92	28.53	29.31
128	28.70	29.63	29.64	30.46	29.12	29.42	29.44	29.23	29.55	29.92
256	29.91	31.95	31.26	31.84	30.21	30.51	30.73	30.32	31.22	32.72

experiment, four images were used to generate the codebook as shown in Fig. 1(b)–(e). Seven images, Fig. 1(f)–(l), were encoded and decoded by the generated codebook. The PSNR values for various images are tabulated in Table 3. From these tables, the generality of the codebook generated by PNM is more robust than the others.

The last three experiments were conducted to compare the algorithms on the codebook initialization, the codeword utility, and the sequence of training vectors problems. Similar to the first three experiments, 10 algorithms were executed for different factors in various codebook sizes. These results are shown in Tables 4–6.

The statistical utility rates of codewords were obtained in various compression rates as listed in Table 4(a). The 'Lena' image was encoded and decoded in this experiment. The compression rates were set as 0.5625, 0.625, and 0.6875 BPP. From this table, the PNM algorithm generated more highly utilized codewords than the other algorithms in various rates. The codebook generated by the PNM

Table 3
The PSNR values for various images encoded by the codebook generated from images 'Lena', 'Pepper', 'F-16', 'Sailboat' and 'Tiffany'

Size	LBG	ELBG	GVQ	GSAVQ	C-ART	CAL	ILBG	ISM	MD	PNM
(a) *Image* '*Goldhill*'										
32	24.22	26.28	26.59	26.63	24.87	25.07	25.84	25.01	26.01	27.27
64	26.28	27.35	27.21	27.52	26.72	26.85	26.95	26.82	27.21	28.44
128	27.40	28.43	28.34	28.66	27.24	27.46	28.01	27.34	28.23	29.36
256	29.01	31.19	31.06	31.15	29.39	29.51	30.54	29.53	31.01	31.51
(b) *Image* '*Toys*'										
32	24.92	27.45	26.89	27.13	25.12	25.52	26.57	25.45	26.85	27.95
64	26.55	28.59	27.85	28.22	27.05	27.35	27.54	27.31	27.81	29.35
128	27.81	29.43	28.95	29.06	28.11	28.45	28.75	28.43	28.89	30.53
256	29.51	31.45	30.86	31.14	29.95	30.35	30.56	30.31	30.84	32.41
(c) *Image* '*Zelda*'										
32	22.72	25.35	24.89	24.93	23.12	23.42	24.35	23.41	24.85	25.82
64	24.35	26.39	25.72	25.92	24.85	25.15	25.51	25.01	25.69	27.25
128	25.41	27.23	26.64	27.06	25.91	26.22	26.45	26.12	26.59	28.35
256	27.31	29.32	28.86	29.04	27.81	28.15	28.56	28.06	28.84	30.25
(d) *Image* '*Girl*'										
32	25.32	27.38	27.25	27.33	25.62	26.02	26.94	25.95	27.22	27.88
64	27.18	28.45	28.31	28.49	27.35	27.75	27.97	27.71	28.28	28.91
128	28.30	29.23	29.24	30.06	28.72	29.02	29.15	28.94	29.21	30.52
256	29.51	31.45	30.86	31.34	29.85	29.91	30.45	29.89	30.83	31.82
(e) *Image* '*Couple*'										
32	25.44	27.45	27.55	27.63	25.72	26.22	27.01	26.19	27.34	27.68
64	27.25	28.55	28.61	28.82	27.45	27.95	28.25	27.92	28.42	28.95
128	28.35	29.43	29.54	30.56	28.82	29.22	29.26	29.12	29.31	30.92
256	29.55	31.65	30.96	31.54	29.95	30.01	30.51	29.98	30.92	32.12
(f) *Image* '*Baboon*'										
32	24.35	26.32	26.64	26.81	24.77	25.27	25.94	25.24	26.25	27.25
64	26.30	27.34	27.40	27.52	26.73	26.95	27.08	26.91	27.23	28.24
128	27.45	28.55	28.42	28.61	27.34	27.66	28.12	27.64	28.37	29.46
256	29.12	31.25	31.24	31.54	29.49	29.91	30.89	29.85	31.16	31.96
(g) *Image* '*Aerial*'										
32	24.55	26.62	26.94	27.01	24.97	25.37	26.12	25.34	26.58	27.45
64	26.50	27.74	27.80	27.92	26.93	27.15	27.35	27.12	27.69	28.44
128	27.65	28.85	28.92	29.11	27.64	27.86	28.24	27.84	28.76	29.86
256	29.42	31.45	31.54	31.84	29.79	30.11	31.02	30.01	31.38	32.16

comprises fewer low-utilized codewords. In order to show the invariance of the initial codebook, 30 initial codebooks were randomly initialized. Image 'Lena' was used to generate the encoding codebooks by 10 algorithms. The PSNR values for 30 initial conditions were averaged in various codebook sizes as tabulated in Table 4(b). Similarly, various sequences of training vectors of image 'Lena' were randomly generated to test the invariance of sample sequence. The variance of averaging PSNR values for these algorithms are shown in Table 4(c). In general, five images 'Lena', 'Pepper', 'F-16', 'Sailboat', and 'Tiffany' were encoded and decoded to perform an inside test as tabulated in Table 5. Seven images 'Goldhill', 'Toys', 'Zelda', 'Girl', 'Couple', 'Baboon' and 'Aerial' were decoded using the codebook generated from the previous five training images. The results of this outside test are tabulated in Table 6 for different factors. From these three tables, the proposed approach has the smallest variance of PSNR values. Our proposed algorithm can be proven to be superior to the others. Additionally, the training time for various algorithms is shown in Fig. 2. Initially, the performance

Table 4
The comparisons for 10 algorithms for different factors encoding and decoding the image 'Lena'

BPP	LBG	ELBG	GVQ	GSAVQ	C-ART	CAL	ILBG	ISM	MD	PNM
(a) *The utility rates in various compression rates*										
0.5625	81.50	96.67	96.57	96.58	85.45	87.65	90.51	86.15	96.45	97.52
0.625	78.58	95.45	95.51	95.61	81.88	83.58	89.15	82.56	95.28	96.12
0.6875	76.84	94.55	94.12	94.34	79.65	81.75	88.54	80.43	94.05	95.24
(b) *The variance of PSNR values using randomly initialized codebooks*										
Size										
32	4.51	1.58	1.15	1.05	4.11	4.01	2.15	2.57	1.65	0.95
64	4.12	1.33	1.12	0.95	3.82	3.72	2.06	2.24	1.54	0.83
128	3.85	1.15	0.93	0.87	3.55	3.35	1.95	2.13	1.34	0.72
256	3.53	0.91	0.85	0.76	3.23	3.13	1.56	2.06	1.23	0.65
(c) *The variance of PSNR values using different sequences of training vectors*										
32	4.21	1.28	1.04	1.01	4.01	3.85	2.21	2.68	1.78	0.96
64	4.02	1.13	1.01	0.89	3.52	3.02	2.18	2.34	1.65	0.82
128	3.55	1.05	0.85	0.81	3.21	2.95	2.01	2.24	1.54	0.70
256	3.23	0.81	0.78	0.71	3.03	2.85	1.72	2.21	1.35	0.61

BPP: Bits per pixel, %.

Table 5
The comparisons of an inside test for different factors

BPP	LBG	ELBG	GVQ	GSAVQ	C-ART	CAL	ILBG	ISM	MD	PNM
(a) *The utility rates in various compression rates*										
0.5625	81.45	96.62	96.51	96.52	85.39	87.59	90.45	86.11	96.39	97.47
0.625	78.52	95.41	95.46	95.55	81.82	83.51	89.09	82.52	95.22	96.07
0.6875	76.78	94.49	94.07	94.28	79.61	81.69	88.49	80.37	94.01	95.19
(b) *The variance of PSNR values using randomly initialized codebooks*										
Size										
32	4.54	1.62	1.18	1.09	4.15	4.05	2.18	2.61	1.69	0.98
64	4.15	1.36	1.16	0.99	3.86	3.76	2.09	2.28	1.58	0.87
128	3.88	1.19	0.97	0.91	3.58	3.38	1.98	2.17	1.38	0.76
256	3.57	0.95	0.89	0.79	3.27	3.17	1.61	2.11	1.26	0.69
(c) *The variance of PSNR values using different sequences of training vectors*										
32	4.26	1.34	1.09	1.06	4.07	3.92	2.27	2.74	1.83	1.01
64	4.07	1.18	1.07	0.95	3.58	3.07	2.24	2.39	1.71	0.88
128	3.61	1.11	0.91	0.86	3.27	3.01	2.07	2.31	1.61	0.76
256	3.29	0.87	0.84	0.77	3.09	2.91	1.78	2.27	1.41	0.67

BPP: Bits per pixel, %.

of the proposed approach was similar to that of CNN-ART. The PSNR value was increased after the MS and PCA modules. After 350 s, the PSNR value of the proposed approach was the highest one.

5. Conclusions

In this paper, a novel algorithm has been proposed to find the better solution for VQ. Four problems in the clustering process have been solved using the NN-based clustering, the MS-based refinement, and the PCA-based seed re-initialization modules. These three modules were repeatedly performed to obtain the better codebook. Some experiments were conducted to make a comparison with the other algorithms on the codebook initialization, the found solution, the codeword utility, and the sequence of samples problems. From the experimental results, the proposed algorithm performed better than the others under the PSNR criterion.

Table 6
The comparisons of an outside test for different factors

BPP	LBG	ELBG	GVQ	GSAVQ	C-ART	CAL	ILBG	ISM	MD	PNM
(a) *The utility rates in various compression rates*										
0.5625	81.39	96.55	96.44	96.45	85.32	87.51	90.38	86.04	96.32	97.41
0.625	78.46	95.34	95.39	95.48	81.75	83.45	89.01	82.44	95.15	96.01
0.6875	76.71	94.42	94.01	94.21	79.54	81.62	88.42	80.31	93.94	95.11
(b) *The variance of PSNR values using randomly initialized codebooks*										
Size										
32	4.59	1.68	1.24	1.15	4.21	4.11	2.24	2.67	1.75	1.03
64	4.21	1.42	1.21	1.05	3.92	3.82	2.15	2.33	1.65	0.93
128	3.93	1.25	1.03	0.96	3.63	3.44	2.04	2.23	1.45	0.81
256	3.62	0.99	0.94	0.84	3.33	3.24	1.66	2.17	1.32	0.75
(c) *The variance of PSNR values using different sequences of training vectors*										
32	4.32	1.41	1.15	1.12	4.13	3.97	2.32	2.81	1.88	1.07
64	4.13	1.24	1.13	1.01	3.65	3.13	2.31	2.45	1.77	0.94
128	3.67	1.17	0.97	0.92	3.34	3.07	2.12	2.37	1.68	0.82
256	3.35	0.93	0.91	0.83	3.15	2.96	1.84	2.33	1.47	0.74

BPP: Bits per pixel, %.

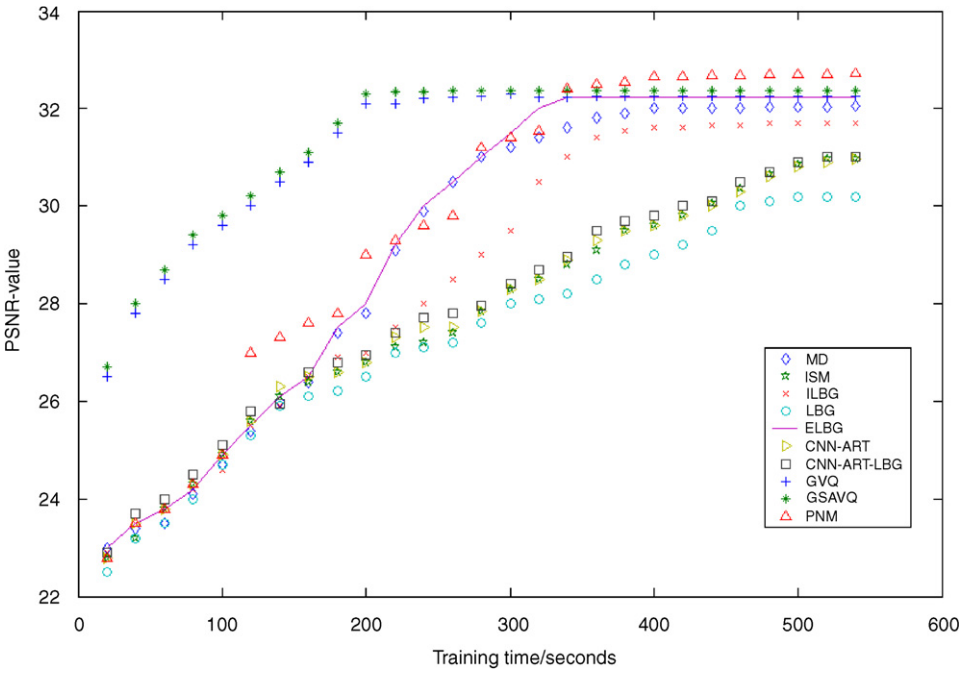

Fig. 2. The comparison of execution time for various algorithms.

Acknowledgment

The work was supported by National Science Council of Taiwan under Grant nos. NSC93-2213-E-239-020 and NSC94-2213-E-239-012.

References

[1] Y. Linde, A. Buzo, R.M. Gray, An algorithm for vector quantizer design, IEEE Trans. Commun. 28 (1980) 84–95.
[2] G. Patane, M. Russo, The enhanced LBG algorithm, Neural Networks 14 (2001) 1219–1237.

[3] T. Kaukoranta, P. Franti, O. Nevalainen, Iterative split-and-merge algorithm for vector quantization codebook generation, Optical Eng. 10 (1998) 2726–2732.
[4] J. Haber, H.-P. Seidel, Using an enhanced lbg algorithm to reduce the codebook error in vector quantization, in: Proceedings of the IEEE International Computer Graphics Conference, 2000, pp. 99–104.
[5] H.-C. Huang, J.-S. Pan, Z.-M. Lu, S.-H. Sun, H.-M. Hang, Vector quantization based on genetic simulated annealing, Signal Processing 81 (2001) 1513–1523.
[6] T.-C. Lin, P.-T. Yu, Centroid neural network adaptive resonance theory for vector quantization, Signal Processing 83 (2003) 649–654.
[7] A. Laha, N.R. Pal, B. Chanda, Design of vector quantizer for image compression using self-organizing feature map and surface fitting, IEEE Trans. Image Process. 13 (2005) 1291–1303.
[8] A. Buzo, A.H. Gray, R.M. Gray, J.D. Markel, Speech coding based upon vector quantization, IEEE Trans. Acoust. Speech Signal Process. ASSP-92 (1980) 562–574.
[9] C.-C. Chang, I.-C. Lin, Novel full-search schemes for speeding up image coding using vector quantization, Real Time Imaging 10 (2004) 95–102.
[10] C.K. Chan, C.K. Ma, A fast method of designing better codebooks for image vector quantization, IEEE Trans. Comm. 42 (1994) 237–242.
[11] Z. Pan, K. Kotani, T. Ohmi, Fast encoding method for vector quantization using modified L_2-norm pyramid, IEEE Signal Process. Lett. 12 (2005) 609–612.
[12] P.-Y. Chen, An efficient prediction algorithm for image vector quantization, IEEE Trans. Systems Man Cybernet. 34 (2004) 740–746.
[13] Y.-L. Huang, R.-F. Chang, A fast finite-state algorithm for generating RGB palettes of color quantized images, J. Inform. Eng. 20 (2004) 771–782.
[14] G.A. Carpenter, S. Grossberg, The ART of adaptive pattern recognition by a self-organization neural network, Computer 21 (1988) 77–88.
[15] D. Commaiciu, P. Meer, Mean shift: a robust approach toward feature space analysis, IEEE Trans. Pattern Anal. Machine Intell. 24 (2002) 603–619.
[16] M.T. Hagan, H.B. Demuth, M.H. Beale, Neural Network Design, Thomson Learning, 1996.