2004－CVIM－145　(12)
2004／9／10

A Self-controlled Incremental Method for Vector Quantization

Shen Furao † Osamu Hasegawa †‡

†Tokyo Institute of Technology
R2-52 4259 Nagatsuta, Midori-ku, Yokohama, 228-8503 Japan
‡PRESTO, Japan Science and Technology Agency (JST)

furaoshen, hasegawa@isl.titech.ac.jp

A new vector quantization method is proposed which generates codebooks incrementally. New vectors are inserted in areas of the input vector space where the quantization error is highest until the desired error threshold is reached. After the desired error threshold is reached, a remove-insert phase fine tunes the codebook. The proposed method can (1) solve the main shortcoming of traditional vector quantization LBG algorithm: the dependence on initial conditions; (2) work better than some recently published efficient algorithm such as Enhanced LBG (Patane 2001) for the traditional task: with fixed number of codewords, to find a suitable codebook to minimize distortion error. (3) work for new task that is not solved with traditional methods: with fixed distortion error, to minimize the number of codewords and find a suitable codebook. By solving some image compression problems, a comparison with ELBG was performed. The results indicate that the new method is significantly better than ELBG.

ベクトル量子化のためのコードブックの自律的・連続的更新法

申　富饒 †　　長谷川　修 †‡

† 東京工業大学
〒 226-8503 横浜市緑区長津田町 4259 (R2-52)
‡‡ 科学技術振興機構　さきがけ研究 21
furaoshen, hasegawa@isl.titech.ac.jp

本稿では、コードブックを自律的かつ連続的に更新する新しいベクトル量子化法を提案する。提案手法は、量子化誤差が最大となる箇所に、誤差が閾値に達するまで新たなベクトルを挿入するアプローチを基本とし、誤差が閾値に達した後は、ベクトルの削除と挿入を繰り返してコードブックをチューニングする。提案手法のポイントは、以下の３点にまとめられる。(1) 主要なベクトル量子化手法のひとつとして知られる LBG の主要な欠点である、「初期状態への依存」の問題を解決した。(2) LBG の改良手法である拡張 LBG (Patane 2001) よりも良好に機能する。すなわち、提案手法は「コードワード数固定」の条件下で誤差を最小にする最適なコードブックを生成可能である。(3) 従来実現されていなかった新たなタスクの実行が可能である。すなわち、提案手法は「誤差固定」の条件下においても、コードワード数を最小にする最適なコードブックを生成可能である。提案手法を画像の圧縮問題に適用し、拡張 LBG との性能比較を行ったところ、提案手法は拡張 LBG に比べ良好な性能を示した。

1 Introduction

The purpose of vector quantization (VQ) [1] is to encode data vectors in order to transmit them over a digital communications channel (this includes data storage/retrieval). Compression via VQ is appropriate for applications where data must be transmitted (or stored) with high bandwidth but tolerating some loss in fidelity. Applications in this class are often found in speech and image processing.

Creating a complete vector quantization system requires both the design of an encoder (quantizer) and a decoder. The input space of the vectors to be quantized is partitioned into a number of disjoint regions. For each region a prototype or output vector (usually called codeword) is found. When given an input vector, the encoder produces the index of the region where the input vector lies. This index, called a channel symbol, can then been transmitted over a binary channel. At the decoder, the index is mapped to its corresponding output vector (codeword). The transmission rate is dependent on the number of quantization regions. Given the number of regions, the task of designing a vector quantizer system is to determine the regions and codewords that minimize the distortion error.

A popular vector quantization method that minimizes the distortion error is the LBG [2] algorithm that is also known as k-means. However, the LBG algorithm is a local search procedure and it is well known that it suffers from the serious drawback that its performance heavily depends on the initial starting conditions. Many works have been developed in order to solve this problem. For example, A. Likas etc. [3] proposed a global k-means algorithm that is an incremental approach to clustering that dynamically adds one cluster center at a time through a deterministic global search procedure consisting of N (with N being the size of the data set) executions of the k-means algorithm from suitable initial positions. The LBG-U [4] or Enhanced LBG (ELBG) [5] algorithm defines one parameter–utility of a codeword, to overcome the main drawbacks of clustering algorithms: the dependence of initial starting conditions.

In this paper, we present a new algorithm we called self-controlled incremental LBG, it determines the number of codewords according to the desired distortion error limitation. By introducing the competitive mechanism, proposed method removes codewords with no or lowest contribution to error minimization, and inserts new codeword near the codeword that do most contribution to error minimization. This process fine tunes the codebook and makes the proposed algorithm independent of initial starting conditions. For the traditional task of vector quantization, i.e., with predefined number of codewords, to find a suitable codebook, the proposed method can work better than a recently published efficient algorithm ELBG [5]. The proposed method can also work for new task, i.e., with predefined distortion error limitation, to minimize the number of codewords and find a suitable codebook. As far as we know, no k-means based method can work for this target.

2 Vector Quantization

2.1 Definition

A vector quantizer Q is a mapping of l-dimensional vector set $X = \{x_1, x_2, \ldots, x_n\}$ into a finite l-dimensional vector set $C = \{c_1, c_2, \ldots, c_m\}$, where $l \geq 2$ and $m \ll n$. Thus

$$Q : X \longrightarrow C \tag{1}$$

C is called codebook, its elements $c_1, c_2, \ldots, c_m$ are called codewords. Associated with m codewords there is a partition $R_1, R_2, \ldots, R_m$ for X, where

$$R_j = Q^{-1}(c_j) = \{x \in X : Q(x) = c_j\} \tag{2}$$

From this definition the regions defining the partition are non-overlapping (disjoint) and their union is X. A quantizer can be uniquely defined by jointly specifying the output set C and the corresponding partition R_j. This definition combines the encoding and decoding steps as one operation called quantization.

Vector quantizer design consists of choosing a distance function $d(x, c)$ that measures the distance between two vectors x and c. Commonly used distance function is the squared Euclidean distance.

$$d(x, c) = \sum_{i=1}^{l} (x_i - c_i)^2 \tag{3}$$

A vector quantizer is called optimal if for a given value of m it minimizes the distortion error. Generally, Mean Quantization Error (MQE) is used as the measure of distortion error:

$$MQE \equiv \frac{1}{m} \sum_{i=1}^{m} E_i \tag{4}$$

where $E_i = \sum_{j:x_j \in R_i} d(x_j, c_i)$ is local distortion error of c_i. Note that the vector quantizer minimizing this Mean Quantization Error is designed to optimally quantize (compress) a given finite vector set.

There are two necessary conditions for an optimal vector quantizer, called the Lloyd-Max conditions [6][7].

1. The codewords c_j must be given by the centroid of R_j:

$$c_j = \frac{1}{N_j} \sum_{i=1}^{N_j} x_i, \quad x_i \in R_j \tag{5}$$

where N_j is the total number of vectors belong to R_j.

2. The partition R_j, $j = 1, \ldots, m$ must satisfy

$$R_j \supset \{x \in X : d(x, c_j) < d(x, c_k) \ \ \forall k \neq j\} \tag{6}$$

We call this partition Voronoi partition.

Note that two necessary conditions can be generalized for any distance function. In that case the output points are determined by the generalized centroid, which is the center of mass determined using a special distance function. The Voronoi partition is also determined using that special distance measure.

The second condition implies that an optimal quantizer must have a Voronoi partition. In that case the quantization regions are defined in terms of the codewords, so the quantizer can be uniquely characterized only in terms of its codewords.

2.2 LBG Algorithm

An algorithm for scalar quantizer was proposed by Lloyd [6], and later Linde, Buzo and Gray [2] generalized it for vector quantization. This algorithm is known as LBG or generalized Lloyd algorithm (GLA). It applies the two necessary conditions to training data in order to determine empirically optimal vector quantizers.

Given training data $x_i, i = 1, \ldots, n$, distance function d, and initial codewords $c_j(0), j = 1, \ldots, m$, the LBG iteratively applies two conditions to produce improved codebook with the following algorithm:

Algorithm 2.1: LBG algorithm

1. Partition the training data $x_i, i = 1, \ldots, n$ into the channel symbols using the minimum distance rule. This partitioning is stored in a $n \times m$ indicator matrix S whose elements are defined by

$$s_{ij} = \begin{cases} 1 & \text{if } d(x_i, c_j(k)) = \min_p d(x_i, c_p(k)) \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

2. Determine the centroids of the training points by channel symbol. Replace the old codewords with these centroids:

$$c_j(k+1) = \frac{\sum_{i=1}^{n} s_{ij} x_i}{\sum_{i=1}^{n} s_{ij}}, \quad j = 1, \ldots, m \tag{8}$$

3. Repeat steps 1-2 until no $c_j, j = 1, \ldots, m$ changes anymore.

Note that the two conditions (formula (5) and (6)) only give necessary conditions for an optimal VQ system. Hence the LBG solution is only locally optimal and may not be globally optimal. The quality of this solution depends on the choice of initial codebook.

3 Self-controlled incremental LBG

3.1 Considerations about LBG algorithm

The LBG algorithm requires initial values for codewords $c_j, j = 1, \ldots, m$. The quality of the solution depends on this initialization. Obviously, if the initial values are near an acceptable solution, there is a better chance that the algorithm will find an acceptable solution. However, poorly chosen initial conditions for codewords lead to "bad" locally optimal solutions.

Enhanced LBG [5] gave a detail analysis of the poorly chosen initial conditions. Fig.1 is an example of badly positioned codewords. If an initial codeword is generated as an empty cell (codeword number 4 in Fig.1), because all the elements of the data set are nearer to the other codewords, following the steps of the traditional LBG, the codeword number 4 cannot move and will never represent any elements. We say the codeword number 4 is useless, it does no contribution to reduction of distortion error. Another problem is that in initial stage, there are too many codewords generated for small cluster but few codewords generated for large cluster. In Fig.1, in the small cluster there are two codewords (codeword number 2 and number 3), in the large cluster, only one (codeword number 1). Even the elements in the smaller cluster are well approximated by the related codewords, a lot of elements in the larger one are badly approximated. We say the number 2 and number 3 codewords do small contribution for reduction of distortion error, and codeword number 1 does large contribution for reduction of distortion error.

In signal processing, this problem is usually cured by applying LBG algorithm many times, starting with different initial conditions and then choosing the best solution. LBG-U [4] and ELBG [5] define different utility parameters to realize a similar target, they try to identify codewords which do not contribute much to the reduction of distortion error and move them to somewhere near the codewords which do much contribution to the reduction of distortion error.

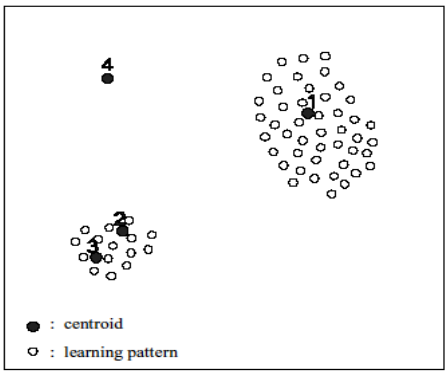

Fig.1: Poorly initialized codewords

In general, with a fixed number of codewords, VQ tries to find a codebook to minimize the distortion error. However, with a fixed distortion error limitation, how to minimize the number of codewords and generate an optimal codebook is also an interesting research topic. With a fixed distortion error, less number of codewords means the higher compression ratio. For example, for image processing, if we want to transmit or store images with same distortion error, the number of codewords is different for different images.

In section 3.2, we propose a so-called self-controlled incremental LBG algorithm. It can work like traditional LBG, i.e., with a fixed number of codewords, to find a codebook to minimize the distortion error. It can also work for new requirements, i.e., with a fixed distortion error limitation, to minimize the number of codewords and generate a suitable codebook.

3.2 Proposed method

As discussed in Section 3.1, the targets of the proposed method are:

- To solve the problem caused by poorly chosen initial conditions as shown in Fig.1.
- With fixed number of codewords, to find a suitable codebook to minimize distortion error. It can work better than some recently published efficient algorithm such as ELBG [5].
- With fixed distortion error, to minimize the number of codewords and find a suitable codebook.

Algorithm 3.1 gives the outline of proposed method.

Algorithm 3.1: Outline of Self-controlled incremental LBG

1. Initialize codebook C to contain one codeword; define the distance function d.
2. Do LBG algorithm (Algorithm 2.1) for codebook C.
3. If insertion condition is satisfied, insert a new codeword to codebook C, then go to step2. Else, if the termination condition is satisfied, output the codebook and stop the algorithm; if the termination condition is not satisfied, do step4.
4. Delete the codewords with no or lowest contribution to reduction of distortion error, go to step2.

In Algorithm 3.1, we need to define the distance function, give insertion condition and termination condition, and design methods of how to insert and delete codewords.

- Distance function. We expect the distance between samples and the center of Voronoi region to be significantly less than the distance between centers of Voronoi region, i.e., within-cluster distance to be significantly less than between-cluster distance. Generally, the Euclidean distance is used as a measure of distance. This particular choice is justified if the feature space is isotropic and the data are spread roughly evenly along all directions. For some particular applications (i.e., speech and image processing), more specialized distance functions exit [8]. In Section 4, we design an adaptive distance function for image compression; this distance function can work better than Euclidean distance when the number of codewords is very large.
- Insertion condition. According to different tasks, different insertion conditions are adopted.
  1. With fixed number of codewords, to minimize the distortion error. The insertion condition is, if the current number of codewords is less than the predefined fixed number of codewords, a new codeword will be inserted to the codebook.
  2. With fixed distortion error, to minimize the number of codewords. The insertion condition is, if the current distortion error is greater than the predefined distortion error, a new codeword will be inserted. Note that lots of distortion error measure can be used here. For example, for image compression, the Peak Signal to Noise Ratio (PSNR) can be used as the measure of distortion error.
- Insertion criteria. To insert a new codeword, we must avoid the bad situation like codeword number 2 and number 3 in Fig.1. Thus, new codeword is inserted near the codeword with highest local distortion error (we call it *winner*). In Fig.1,

codeword number 1 is the *winner*, and we randomly choose a vector lies in the Voronoi region of codeword number 1 as a new codeword.

- Deletion criteria. If the local distortion error of a codeword is 0 or the lowest compared with other codewords, we call the codeword *loser*. The *loser* will be deleted. The codeword adjacent to the *loser* accepts the Voronoi region of *loser*. For example, in Fig.1, if we delete codeword number 2, all vectors in the Voronoi region R_2 are assigned to Voronoi region R_3, and codeword number 3 is moved to the centroid of the new region.

- Termination condition. We hope the deletion and insertion of codewords is able to fine tune the codebook. It means that, for fixed number of codewords, deletion and the following insertion of codewords will lead to decrease of distortion error; for fixed distortion error limitation, deletion and the following insertion of codewords will lead to decrease of number of codewords without increase of distortion error. If the condition is not satisfied, the algorithm outputs the generated codebook and stop.

With the above analysis, we give the whole proposed method in Algorithm 3.2.

Notations used in Algorithm 3.2

- C: codebook.
- c_i: ith codeword of codebook C.
- E_i: local distortion error of codeword c_i.
- R_i: Voronoi region of codeword c_i.
- m: predefined total number of codewords.
- η: predefined total distortion error limitation.
- q: current number of codewords.
- ξ: current total distortion error.
- iq: inherited number of codewords. It stores the number of codewords when total distortion error ξ is less than η.
- $i\xi$: inherited distortion error. It stores the total distortion error when number of codewords q is equal to m.
- iC: inherited codebook. It stores the codebook when ξ is less than η (or q is equal to m).
- *winner*: the codeword with largest local distortion error.
- *loser*: the codeword whose local distortion error is 0 or lowest.

Algorithm 3.2: Self-controlled incremental LBG

1. Initialize the codebook C to contain one codeword c_1

$$C = \{c_1\} \tag{9}$$

with codeword c_1 chosen randomly from original data vectors set. Predefine the total number of codewords as m (or the distortion error limitation as η), give the definition of distance function d, and initialize inherited number, distortion error, and codebook as:

$$iq = +\infty \tag{10}$$
$$i\xi = +\infty \tag{11}$$
$$iC = C \tag{12}$$

2. With codebook C, do LBG algorithm (Algorithm 2.1) to optimize codebook C. Record the current number of codewords q, current total distortion error ξ, every local distortion error E_i, and Voronoi region R_i of c_i $(i = 1, \ldots .q)$.

3. (a) If current number of codewords q is less than m (or current distortion error ξ is greater than η), insert a new codeword c_{new} to codebook C, i.e., if $q < m$ (or $\xi > \eta$), do the following to insert a new codeword:
   - Find the *winner* whose local distortion error is largest.

$$C_{winner} = \arg\max_{c_i \in C} E_i \tag{13}$$

   - Randomly choose one vector c_{new} from the Voronoi region R_{winner} of c_{winner}.
   - Add the c_{new} to codebook C.

$$C = C \cup c_{new} \tag{14}$$

   Then go to step2 to do LBG with the new codebook C.

   (b) If current number of codewords q is equal to m (or current distortion error ξ is less than or equal to η), i.e., if $q = m$ (or $\eta \geq \xi$), do the following:
   - If $\xi < i\xi$ (or $q < iq$),

$$i\xi = \xi \tag{15}$$
$$iq = q \tag{16}$$
$$iC = C \tag{17}$$

   go to step4.

- If $\xi \geq i\xi$ (or $q \geq iq$), output iC (codebook), iq (number of codewords), and $i\xi$ (distortion error) as the final results, stop.

4. Do following steps to delete codewords with 0 or lowest local distortion error.

   - Find the *loser* with $E_{loser} = 0$ or

   $$c_{loser} = arg \min_{c_i \in C, E_i \neq 0} E_i \tag{18}$$

   - Delete c_{loser} from codebook C.

   $$C = C \backslash c_{loser} \tag{19}$$

   - Find the codeword c_a adjacent to c_{loser}, assign all vectors in R_{loser} to R_a, substitute c_a by the centroid of new Voronoi region.

   $$R_a^{'} = R_a \cup R_{loser} \tag{20}$$

   $$c_a^{'} = \frac{1}{N_a} \sum_{i=1}^{N_a} x_i, \quad x_i \in R_a^{'} \tag{21}$$

   Then go to step2 to do LBG with the new codebook C.

4 Experiment

In this section we will examine proposed algorithm with several image compression tasks. We will compare our results with the recently published efficient algorithm ELBG [5]. In the experiments, three images Lena ($512 \times 512 \times 8$), Boat ($512 \times 512 \times 8$), and Gray21 ($512 \times 512 \times 8$) are tested. To find suitable codebooks of such images, we give the scheme of distance function, measure of distortion error, and how to get vector set from images.

- Distance function. An adaptive distance function is defined as follows: Assume the current number of codewords is q, the distance function $d(x, c)$ is defined as:

$$p = \log_{10} q + 1 \tag{22}$$

$$d(x, c) = (\sum_{i=1}^{l} (x_i - c_i)^2)^p \tag{23}$$

From this definition we know, following the insertion of new codewords, the number of codewords becomes larger and larger, and the measure of distance becomes larger and larger. It is because if we use the definite measure of distance, following the increasing of number of codewords, the distance between codewords, i.e., the distance between the center of Voronoi regions, becomes less and less. Thus, when the number of codewords is large enough, the distance between samples and codewords cannot be significantly less than the distance between centers. With the adaptive distance function (formula (23)), following the increasing of number of codewords, the measure of distance also increased. This technique make sure that the within cluster distance is significantly less than the between cluster distance.

- Measure of distortion error. In image applications, often, the Peak Signal to Noise Ratio (PSNR) is used to evaluate the resulting images after the quantization process. The PSNR is defined as follows:

$$PSNR = 10 \log_{10} \frac{255^2}{\frac{1}{N} \sum_{i=1}^{N} (f(i) - g(i))^2} \tag{24}$$

where f and g are respectively the original image and the reconstructed one. All grey levels are represented with an integer value comprised in [0, 255]. N is the total number of pixels in image f. From the definition we know, higher PSNR means lower distortion error, and thus the reconstructed image is better. Here, we adopt PSNR as the measure of distortion error.

- Vector data set. To compare the proposed method with ELBG, we adopt the same vector data set as ELBG used. The image is divided into 4×4 non-overlapping blocks and each block is a vector. The resulting 16-dimensional vectors are used as an input vector data set. For example, for Lena's image of 512×512 pixels, there are 16384 16-dimensional vectors in the data set.

In [5], G. Patane and M. Russo proposed an enhanced LBG (ELBG) algorithm to do vector quantization. They designed a parameter - utility of a codeword, to overcome one of the main drawbacks of clustering algorithms: generally, the results achieved are not good in the case of a bad choice of the initial codebook. They do some experiments to show that ELBG is able to find better codewords than previous clustering techniques. In this paper, we use ELBG as the benchmark algorithm to show the advantage of the proposed method. We do three experiments to examine the proposed method and compare the results with ELBG.

4.1 Experiment 1

With fixed number of codewrods, to generate a suitable codebook and maximize the PSNR (thus minimize the

distortion error). The test image is Lena ($512 \times 512 \times 8$) (Fig.2). According to Table 6 of [5], ELBG is tested with 256, 512, and 1024 codewords. Here, we also set the predefined number of codewords m as 256, 512, and 1024, then execute Algorithm 3.2 to generate a suitable codebook.

In Experiment 1, step3 of Algorithm 3.2 becomes:

(a) When current number of codewords q is less than m, insert a new codeword and go to step2.

(b) If $q = m$

- If current PSNR $psnr$ is greater than the inherited PSNR i_psnr, i.e., if $psnr > i\_psnr$,

$$i\_psnr = psnr \tag{25}$$
$$iq = q \tag{26}$$
$$iC = C \tag{27}$$

  then go to step4.

- If $i\_psnr \geq psnr$, output iC as the final codebook, stop.

Table1: Experiment 1: with fixed number of codewords, ELBG and Proposed method comparison

Number of codewords	PSNR	
	ELBG	Proposed
256	31.94	31.98
512	33.14	33.18
1024	34.59	34.70

The results of proposed method are listed in Column 3 of Table 1. Fig.3 gives the reconstructed image with 256 codewords. From Table 1 we find that, with the same number of codewords, proposed method gets higher PSNR than ELBG, i.e., proposed method gets better codebook than ELBG.

Fig.2: Original image of Lena ($512 \times 512 \times 8$)

Fig.3: Reconstructed image of Fig.2, Proposed method, $m = 256$, $PSNR = 32.03dB$

4.2 Experiment 2

With fixed PSNR limitation, to minimize the number of codewords and generate a suitable codebook. The test image is Lena ($512 \times 512 \times 8$) (Fig.2). Also considering Table 6 of reference [5], we set the ELBG results 31.94, 33.14, and 34.59 as the PSNR limitation.

Here, the step3 of Algorithm3.2 becomes:

(a) If current PSNR $psnr$ is less than predefined PSNR threshold η_psnr, insert a new codeword and go to step2.

(b) If $psnr \geq \eta\_psnr$

- If current number of codewords q is less than inherited number of codewords iq, then

$$i\_psnr = psnr \tag{28}$$
$$iq = q \tag{29}$$
$$iC = C \tag{30}$$

  go to step4.

- If $q \geq iq$, output iC as the final codebook, stop.

The results of proposed method are listed in Column 3 of Table2. It shows that, with the same PSNR, proposed method needs less number of codeword than ELBG. It means that with the same reconstruction quality (PSNR), proposed method can get higher compression ratio than ELBG.

Table2: Experiment 2: with fixed PSNR, ELBG and Proposed method comparison

PSNR	Number of codewords	
	ELBG	Proposed
31.94	256	247
33.14	512	505
34.59	1024	1001

4.3 Experiment 3

With the same fixed PSNR limitation, to find suitable codebook for different images. The test images are Lena ($512 \times 512 \times 8$) (Fig.2), Gray21 ($512 \times 512 \times 8$) (Fig.4), and Boat ($512 \times 512 \times 8$) (Fig.5). The three images are different in detail. For example, Gray21 is flat and there is little ditail. Lena has more detail than Gray21 but less detail than Boat.

This experiment is to check with the same PSNR limitation, if different images with different detail need different number of codewords. The PSNR threshold η_psnr is set as 28, 30, and 33. The step3 of Algorithm3.2 is the same as Experiment 2.

Table 3 lists the results. It shows that, for different images, if there is less detail in the image, less codewords are needed. Thus, when some image database needs to be compressed with the same distortion error (i.e., the same reconstruction fidelity), the proposed method can be used, and for different images, different compression ratio is obtained.

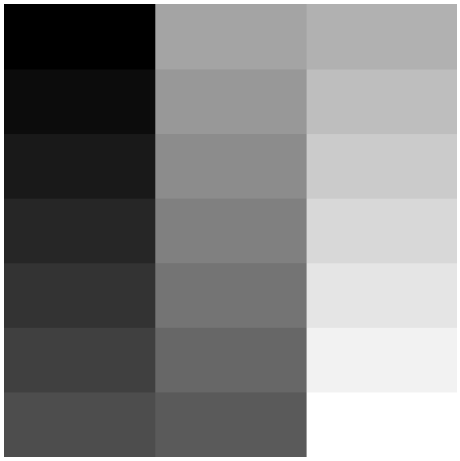

Fig.4: Original image of Gray21 ($512 \times 512 \times 8$)

Fig.5: Original image of Boat ($512 \times 512 \times 8$)

Table3: Experiment 3: with fixed PSNR, Proposed method works for different images

PSNR	Number of codewords		
	Gray21	Lena	Boat
28	9	23	55
30	12	76	199
33	16	462	1042

5 Conclusion

In this paper, we proposed a self-controlled incremental LBG algorithm for vector quantization. New codewords are inserted to codebook until insertion condition is not satisfied, deletion and insertion technique fine tunes the codebook and get better performance than most recently published methods. Proposed method can not only work for traditional VQ tasks (predefine the number of codewords), but can work for new requirements (predefine the distortion error limitation). Experiments for complex image compression tasks also show the efficiency of proposed method. Compared to the most recently published method ELBG, proposed method can get better codebook than ELBG.

References

[1] Vladimir, C., & Filip M. (1997). Learning from data-Concepts, Theory, and Methods. JOHN WILEY & SONS, INC.

[2] Linde, Y., Buzo, A., & Gray, R.M. (1980). An algorithm for vector quantizer design. IEEE Transactions on Communication, COM-28:84-95.

[3] Likas, A., Vlassis, N., & Verbeek, J.J. (2003). The global k-means clustering algorithm. Pattern Recognition, Vol. 36, 451-461.

[4] Fritzke, B. (1997). The LBG-U method for vector quantization - an improvement over LBG inspired from neural networks. Neural Processing Letters, Vol. 5, No. 1.

[5] Patane, G., & Russo, M. (2001). The enhanced LBG algorithm. Neural Networks, Vol. 14, 1219-1237.

[6] Lloyd, S.P. (1957). Least squares quantization in PCM's, Bell Telephone Laboratories Paper, Murray Hill, NJ.

[7] Max, J. (1960). Quantizing for minimum distortion, IRE Trans. Info. Theory, IT-6, 7-12.

[8] Gray, R.M. (1984). Vector quantization. IEEE ASSP Mag., 1, 4-29.