Ain Shams Engineering Journal (2016) xxx, xxx–xxx

Ain Shams University

Ain Shams Engineering Journal

www.elsevier.com/locate/asej
www.sciencedirect.com

ELECTRICAL ENGINEERING

Image compression based on vector quantization using cuckoo search optimization technique

Karri Chiranjeevi *, **Uma Ranjan Jena**

Department of Electronics and Tele-communication Engineering, Veer Surendra Sai University of Technology (VSSUT), Burla 768018, Odisha, India

Received 17 August 2015; revised 14 July 2016; accepted 8 September 2016

KEYWORDS

Cuckoo search (CS);
Firefly algorithm (FA);
Particle swarm optimization (PSO);
Linde-Buzo-Gray (LBG);
Vector quantization;
Image compression

Abstract Most common vector quantization (VQ) is Linde Buzo Gray (LBG), that designs a local optimal codebook for image compression. Recently firefly algorithm (FA), particle swarm optimization (PSO) and Honey bee mating optimization (HBMO) were designed which generate near global codebook, but search process follows Gaussian distribution function. FA experiences a problem when brighter fireflies are insignificant and PSO undergoes instability in convergence when particle velocity is very high. So, we proposed Cuckoo search (CS) metaheuristic optimization algorithm, that optimizes the LBG codebook by levy flight distribution function which follows the Mantegna's algorithm instead of Gaussian distribution. Cuckoo search consumes 25% of convergence time for local and 75% of convergence time for global codebook, so it guarantees the global codebook with appropriate mutation probability and this behavior is the major merit of CS. Practically we observed that cuckoo search algorithm has high peak signal to noise ratio (PSNR) and better fitness value compared to LBG, PSO-LBG, Quantum PSO-LBG, HBMO-LBG and FA-LBG at the cost of high convergence time.

1. Introduction

Image compression plays a significant role in multimedia applications. Presently establishment of image compression techniques with excellent reconstructed image quality is a crucial and challenging task for researchers. Image compression is aimed to transmit the image with lesser bits. Identification of redundancies in image, perfect and suitable encoding technique and transformation technique are the main factors for image compression. The primary image compression technique was JPEG [1] introduced by a group called Joint Photographic Expert Group (JPEG). Quantization is of two types: scalar quantization and vector quantization. Vector quantization being a non-transformed compression technique, is a powerful and efficient tool for lossy image compression. The main aim of vector quantization was to design an efficient codebook that contains a group of codewords to which input image vector is assigned based on the minimum Euclidean distance. The primary and most used vector quantization technique is Linde

* Corresponding author.
E-mail addresses: chiru404@gmail.com (K. Chiranjeevi), urjena@rdiffmail.com (U.R. Jena).

Peer review under responsibility of Ain Shams University.

ELSEVIER Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.asej.2016.09.009
2090-4479 © 2016 Ain Shams University. Production and hosting by Elsevier B.V.

Please cite this article in press as: Chiranjeevi K, Jena UR, Image compression based on vector quantization using cuckoo search optimization technique, Ain Shams Eng J (2016), http://dx.doi.org/10.1016/j.asej.2016.09.009

Buzo Gray (LBG) algorithm (in 1980) [2]. LBG algorithm is simple, adaptable and flexible, and is based on the minimum Euclidean distance between the image vector and corresponding codeword. It produces a local optimal solution but, does not guarantee the best global solutions. LBG algorithm final solution depends on initial codebook which is generated randomly. Patane & Russo proposed an enhanced LBG (ELBG) algorithm that improves the local optimal solution of LBG algorithm [3]. The basic idea of ELBG is optimal utility of codewords, a powerful instrument to overcome the main drawbacks of clustering algorithms and the result shows better performance than LBG and its performance is independent on initial codebook. Projection Vector Quantization (PVQ) adopts the quadtree decomposition for segmenting the image into variable sized blocks which are well represented by the single orientation reconstruction (SOR) and it shows improved performance in both subjective and objective qualities compared with the case that uses fixed sized blocks [4]. Object-based VQ is performed in three phases, initialization, iterative and finalization. The initialization phase is based on Max–Min algorithm. The iterative phase is an adaptive LBG algorithm. The finalization frees the codebook from redundancy [5]. A quad tree (QT) decomposition algorithm allows VQ with variable block size by observing homogeneity of local regions [6], but Kazuya et al. (in 2008) observed that complexity of local regions of an image is more essential than the homogeneity [7]. So a vector quantization of images with variable block size quantifying the complex regions of the image using local fractal dimensions (LFDs) is proposed. Dimitrios et al. (in 2012) proposed a fuzzy vector quantization for image compression based on competitive agglomeration and a novel codeword migration strategy [8]. Dimitrios et al. proposed a learning mechanism to systematically design fast fuzzy clustering-based vector quantizers by combining three learning modules [9]. However, a multivariate vector quantization (MVQ) approach is used for compression of hyperspectral imagery (HSI). His effective codebook is designed by using the fuzzy C-mean (FCM). Results shows that proposed MVQ outperforms conventional VQ in mean square error (MSE) and reconstructed image quality [10]. Wang and Meng observed that image compression can also be performed with transformed vector quantization in which image to be quantized is transformed with discrete wavelet Transform (DWT) [11].

In the recent past soft computing techniques have developed in the fields of engineering and technological problems. Rajpoot et al. designed a codebook by using an ant colony optimization (ACO) algorithm [12]. They designed a codebook using ACO by representing the wavelet coefficient in a bidirectional graph and defining a suitable mechanism for placing edges on the graph and proved better than LBG but, convergence time is high. So Tsaia et al. proposed a fast ant colony optimization for codebook generation by observing the redundant calculations in ACO algorithm [13]. They improved the speed of convergence of ACO by identifying the redundant calculations in designing a codebook and proved better than ordinary ACO. In addition, particle swarm optimization vector quantization [14], is based on updating the particle global best (*gbest*) and local best (*pbest*) solutions and it outperforms LBG algorithm. The *gbest* holds highest fitness value among all populations and *pbest* holds the best fitness value of corresponding particle. Feng et al. showed that evolutionary fuzzy particle swarm optimization algorithm [15] has better global codebook and performance is better than PSO and LBG algorithms. QPSO was proposed by Wang (2007) to solve the 0–1 knapsack problem [16] and to improve the performance of PSO. The QPSO performance is better than PSO; it computes the local points from the *pbest* and *gbest* for each particle and updates the position of the particle by choosing appropriate parameters u which is a random number lying between 0 and 1 and z is non-negative constant lower than 2.8. Chang et al. proposed a tree structured vector quantization for fast codebook design with the help of employing the triangle inequality to achieve efficient codewords with the cost of high convergence time [17]. So Yu-Chen et al. proposed a fast codebook search algorithm that employs two test conditions to speed up the image encoding procedure without incurring any extra image distortion. According to the results, an average 95.23% reduction in execution time can be achieved when there was a codebook of 256 codewords [18].

Sanyal et al. (2013) applied a new approach in selection of chemotaxis steps of basic Bacterial foraging optimization algorithm (BFOA) which leads the algorithm to develop a near optimal codebook for image compression with good reconstructed image quality and high peak signal to noise ratio [19]. They choose a fuzzy membership functions as a objective function which is optimized by the modified Bacterial foraging optimization and compared the results with other optimization techniques. Horng and Jiang applied honey bee mating optimization algorithm for vector quantization [20]. HBMO has high quality reconstructed image and better codebook with small distortion compared to PSO-LBG, QPSO-LBG and LBG algorithm. Horng (in 2012) applied a firefly algorithm [21] to design a codebook for vector quantization. Firefly algorithm is encouraged by social activities of fireflies and the occurrence of bioluminescent communication. Fireflies with less brighter intensity values move toward the brighter intensity fireflies if there are such. Otherwise they move randomly. The FA undergoes a problem when there are no such brighter fireflies in the search space, so Chiranjeevi et al. proposed a modified FA in which fireflies follow a specific strategy when there are no such brighter fireflies in the search space [22]. Chiranjeevi and Jena applied a bat optimization algorithm for efficient codebook design with appropriate selection of tuning parameters (loudness and frequency) and proved better in PSNR and convergence time than FA [23]. Color images are also compressed with VQ by compressing raw data before demosaicking and perform mosaicking to reconstruct the R, G and B bands. A novel VQ technique for encoding the wavelet decomposed color image using Modified Artificial Bee Colony (ABC) optimization algorithm and results are compared with Genetic Algorithm and ordinary ABC with standard LBG algorithm and results show higher PSNR indicating better reconstruction [24].

In our proposed method, we applied Cuckoo search for the first time to design effective and efficient codebook which results in better vector quantization and leads to a high peak signal to noise ratio (PSNR) with good reconstructed image quality. Cuckoo search algorithm is applicable for maximizing/minimizing many linear and nonlinear problems. In this work, first time we grabbed application of cuckoo search algorithm for efficient and effective codebook design. The parameter setting of PSO, QPSO, HBMO and FA is a crucial task and improper tuning of parameters affects the performance of the algorithm. The CS algorithm, on the other hand is

Please cite this article in press as: Chiranjeevi K, Jena UR, Image compression based on vector quantization using cuckoo search optimization technique, Ain Shams Eng J (2016), http://dx.doi.org/10.1016/j.asej.2016.09.009

simple and involves only two tuning parameters (skewness and mutation probability). For this reason, the algorithm outperforms the existing PSO, QPSO HBMO and FA for linear and nonlinear numerical optimization problems. The CS algorithm needs initialization of number of populations and number of iterations which are common to all algorithms. A balance between exploitation and exploration is important for any optimization technique which affects the performance of technique. The CS algorithm is a global optimizer to explore the search space which gives efficient/optimal codebook. From the above discussion, it arises that the CS algorithm is best in exploitation. In this paper, the performance of CS is compared with the PSO, QPSO, HBMO and FA. The CS is shown to be better in efficient codebook design leading to a good image compression. This paper is organized into five sections including the introduction. In Section 2 recent methods of codebook design are discussed along with their algorithms. The proposed method of CS-LBG algorithm is presented with the procedure in Section 3. The results and discussions are given in Section 4. Finally the conclusion is given in Section 5.

2. Vector quantization

Vector Quantization is carried out in three steps- encoder, channel and decoder. The schematic diagram of Vector Quantization is shown in Fig. 1. The diagram consists of three blocks in which each block has a different working principle. Block 1 is the encoder section which includes generation of image vectors, codebook generation and indexing. Image vectors are generated by subdividing the input image into immediate and non-overlapping blocks. Generation of efficient codebook is the major task in vector quantization. A codebook contains a group of codewords of size equal to non-overlapping block size. An algorithm is said to be better algorithm if its generated codebook is efficient. After a successful generation of the codebook, each vector is indexed with the index number from index table. These index numbers are transmitted to the receiver. Block 2 is the channel through which indexed numbers are transmitted to the receiver. Block 3 is a decoder section which includes index table, codebook and reconstructed image. The received indexed numbers are decoded with receiver index table. The codebook at the receiver is the same as that of transmitter codebook. The received index numbers are assigned to its corresponding codewords and these codewords are arranged in a manner that the size of the reconstructed image is the same as that of the input image.

2.1. LBG vector quantization algorithm

The most commonly used VQ is Generalized Lloyd Algorithm (GLA) also called Linde-Buzo-Gray (LBG) algorithm. The distortion becomes smaller after recursively executing the LBG algorithm, but it provides local codebook. The algorithm is as follows:

Step 1: Begin with initial codebook C_1 of size N. Let the iteration counter be $m = 1$ and the initial distortion $D_1 = \infty$.
Step2: Using codebook $C_m = \{Y_i\}$, partition the training set into cluster sets R_i using the nearest neighbor condition.
Step 3: Once the mapping of all the input vectors to the initial code vectors is made, compute the centroids of the partition region found in step 2. This gives an improved codebook C_{m+1}.
Step 4: Calculate the average distortion D_{m+1}. If $D_m - D_{m+1} < T$ then stops, otherwise $m = m + 1$ and repeat step 2 to step 4.

2.2. PSO-LBG vector quantization algorithm

The PSO was proposed by Kennedy and Eberhart in the year 1995 [25]. It is based on social behavior of bird flocking or fish schooling. There are two categories of PSO models: *gbest* and *lbest* models. Chen et al. [26] used the PSO *gbest* model to design a codebook for vector quantization by initializing the result of a LBG algorithm as *gbest* particle so that it increases the speed of convergence of PSO. In PSO particles/codebooks alter their values based on their previous experience and the

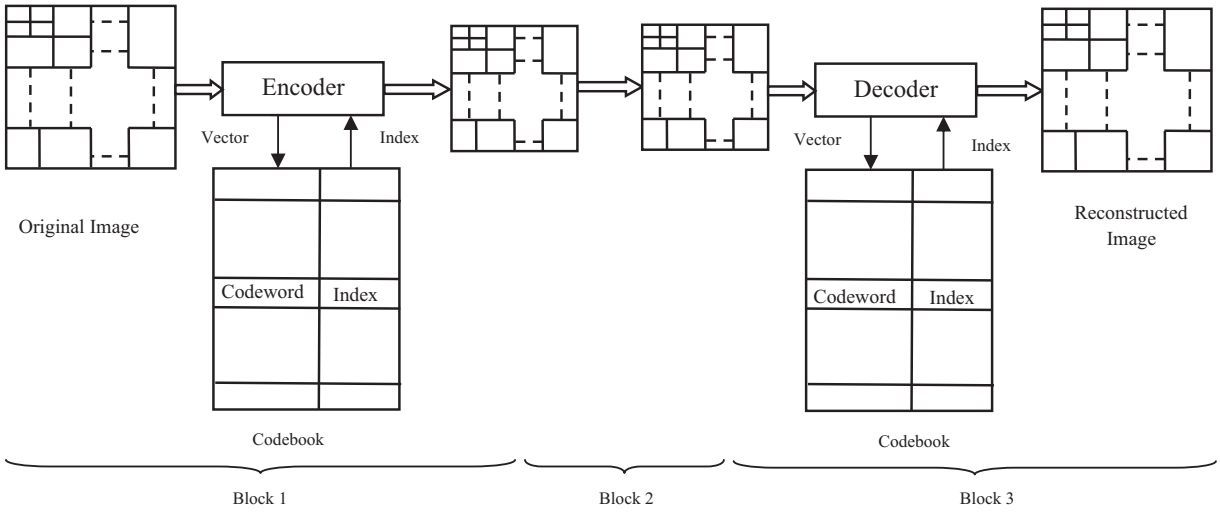


Figure 1 Encoding and decoding process of vector quantization.

best experience of the swarm to generate a best codebook. The structure of codebook for optimization techniques with size N_c and length N_b is shown in Fig. 2. Here codebook is assumed as a particle. For a maximization problem, the quality or fitness of a solution can simply be proportional to the objective function. The PSO provides a near global codebook, but undergoes a problem when the velocity of particle value is maximized. The PSO algorithm follows:

Step 1: Run the LBG algorithm; assign its outcome as global best codebook (*gbest*).
Step 2: Initialize rest codebooks with random numbers and their corresponding velocities.
Step 3: Find out fitness values by Eq. (1) for each codebook.

$$Fitness(C) = \frac{1}{D(C)} = \frac{N_b}{\sum_{j=1}^{N_c}\sum_{i=1}^{N_b} u_{ij} \times \|X_i - C_j\|^2} \quad (1)$$

where X_i is the ith input image vector and C_j is jth codeword of size N_b in a codebook of size N_c and u_{ij} is 1 if X_i is in the jth cluster, otherwise zero.
Step 4: If new fitness value of a codebook is better than old fitness (*pbset*) then assign its corresponding new fitness as pbest.
Step 5: Select the highest fitness value among all the codebooks and if it is better than *gbest*, then replace *gbest* with the selected highest fitness value.
Step 6: Update the velocities by Eq. (2) and update each particle to a new position by Eq. (3) and return to Step 3.

$$v_{ik}^{n+1} = v_{ik}^{n} + c_1 r_1^n \left(pbest_{ik}^n - X_{ik}^n\right) + c_2 r_2^n \left(gbest_k^n - X_{ik}^n\right) \quad (2)$$

$$X_{ik}^{n+1} = X_{ik}^{n} + v_{ik}^{n+1} \quad (3)$$

where k is the number of solutions, i is the position of the particle and c_1, c_2 are cognitive and social learning rates respectively. r_1 and r_2 are random numbers.
Step 7: Until stopping criterion is satisfied (Maximum iteration) repeat step 3 to step 7.

2.3. QPSO-LBG vector quantization algorithm

Unlike PSO, the QPSO computes the local point P_i [8] from the *pbest* and *gbest* for ith codebook according to Eq. (4).

$$P_i = r_1 pbest_i + r_2 gbest_i / r_1 + r_2 \quad (4)$$

Furthermore, two parameters, u and z, are defined to update the position of the particle associated with the local point. To sum up, the three parameters are used to update the particle. Here codebook is assumed as a particle. The detailed algorithm is as follows.
Step 1: Run the LBG algorithm; assign its outcome as global best codebook (*gbest*) and initialize the rest of codebooks and velocities randomly.
Step 2: Find fitness values of all codebooks by Eq. (1).
Step 3: If new fitness value of any codebook is better than old fitness value (*pbset*) then assign its corresponding new fitness as *pbest*.
Step 4: Select the highest fitness value among all the particles and if better than *gbest*, then replace *gbest* with highest fitness value.
Step 5: Select r_1, r_2 and u randomly in range 0–1, further, the local point P_i is calculated by using Eq. (4).
Step 6: Update each element of codebook X_i by Eqs. (5) and (6).

$$L_i = z|X_i - p_i| \quad (5)$$

$$\text{If } u > 0.5 \quad X_i(t+1) = p_i - L_i \cdot \ln(1/u) \quad (6)$$

$$\text{else} \quad X_i(t+1) = p_i + L_i \times \ln(1/u)$$

where z is non-negative constant and is less than $1/\ln\sqrt{2}$ and t is the current iteration time.
Step 7: Repeat steps 3 to step 7 until stopping criteria are satisfied.

2.4. HBMO-LBG vector quantization algorithm

Many algorithms are proposed by the researchers based on the behavior of honey bees [27]. These algorithms are mainly

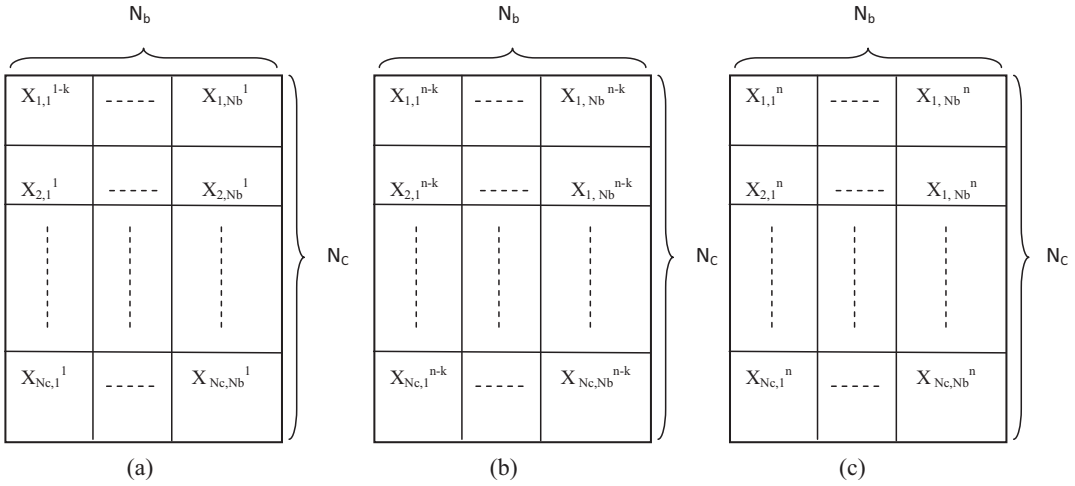


Figure 2 The structures of codebook in optimization techniques (a) the first codebook ($N_c \times N_b$); (b) the $(n-k)^{\text{th}}$ codebook ($N_c \times N_b$); (c) the n^{th} codebook ($N_c \times N_b$).

classified into two categories according to the nature of behavior: foraging behavior and the mating behavior. Karaboga and Basturk [28] proposed the Artificial Bee Colony (ABC) Algorithm based on the foraging behavior of the bees. HBMO is based on mating process of honey bees [29]. Here codebooks are assumed as Bees. The detailed algorithm for vector quantization is as follows.

Step 1: Assign the codebook of LBG algorithm as a queen Q and generate other codebooks (drones) randomly.
Step 2: Calculate the fitness of all the drones by Eq. (1) and assign largest as D_{best}, if the D_{best} is better than the queen Q fitness then queen Q is replaced by D_{best}.
Step 3: Queen Q select a drone for mating if the drone passes the probabilistic condition Eq. (7) then add sperm of the drone in the spermatheca.

$$P_D(t) = e^{\left[\frac{-\Delta(f)}{Speed(t)}\right]} \tag{7}$$

Step 4: The new brood is generated from the queen Q and the sperm by using Eqs. (8–10)

$$brood = Q \pm \beta \times (sperm - Q) \tag{8}$$

$$Speed(t+1) = \alpha \times Speed(t) \tag{9}$$

$$Energy(t+1) = \alpha \times energy(t) \tag{10}$$

where β and α are random numbers between 0 and 1.
Step 5: Select a sperm from the spermatheca and generate a brood by applying a crossover operator between the queen and the selected drones.
Step 6: If the brood's fitness is better than the queen's fitness then replace the queen with the brood else if the brood's fitness is better than one of the drone's fitness then replace the drone with the brood.
Step 7: Repeat step 3 to step 7 until maximum iterations.

2.5. FA-LBG vector quantization algorithm

Firefly algorithm is introduced by Yang (in 2008) [30,31]. The FA is inspired by the flashing pattern and characteristics of fireflies. In this algorithm we are assuming that brightness of a firefly is equal to objective function value. The lower intensity firefly (lower fitness value) moves toward brighter firefly (higher fitness value). Here codebooks are assumed as fireflies. The detailed FA algorithm is given below:

Step 1: Run the LBG algorithm and assign its outcome as brighter codebook.
Step 2: Initialize alpha (α), attractiveness (β_0) and light absorption coefficient (γ) parameters. Initialize rest codebooks with random numbers.
Step 3: Find fitness value of each codebook by Eq. (1).
Step 4: Randomly select a codebook and record its fitness value. If there is a brighter codebook, then it moves toward the brighter codebook (highest fitness value) based on Eqs. (11)–(13).

$$\text{Euclidean distance } r_{ij} = \|X_I - X_J\| = \sqrt{\sum_{k=1}^{N_c}\sum_{h=1}^{L}\left(X_{i,k}^h - X_{j,k}^h\right)^2} \tag{11}$$

Here X_I is randomly selected codebook, X_J is brighter codebook.

$$\beta = \beta_0 e^{-\gamma_{i,j}} \tag{12}$$

$$X_{j,k}^h = (1-\beta)X_{i,k}^h + \beta X_{j,k}^h + u_{j,k}^h \tag{13}$$

where u is a random number between 0 and 1, $k = 1,2,\ldots,N_c$, $h = 1,2,\ldots,L$.
Step 5: If no firefly fitness value is better than the selected firefly then it moves randomly in search space according to Eq. (14)

$$X_{i,k}^h = X_{i,k}^h + u_{j,k}^h \quad k = 1,2,\ldots,N_c,\ h = 1,2,\ldots,L \tag{14}$$

Step 6: Repeat step 3 to step 5 until one of the termination criteria is reached.

3. Proposed CS-LBG vector quantization algorithm

PSO generates an efficient codebook, but undergoes instability in convergence when particle velocity is very high [32]. FA generates near global codebook, but it experiences a problem when there are no such significant brighter fireflies in the search space [33]. The Cuckoo Search algorithm is a nature inspired (behavior and breeding process of cuckoo birds) optimization algorithm developed by Yang and Deb (2009) at Cambridge University [34,35] and is proposed for generation of the global codebook with one tuning parameter. It is applicable for both linear and nonlinear problems. Cuckoo birds emit beautiful sounds and its reproduction approach inspires the researchers. Cuckoo birds lay their eggs in the nests of host birds. If the host bird recognizes those eggs are not of its own, it throws them away or abandons the nest and searches for a nest at any new location [36]. Non-parasitic cuckoos, like most other non-passerines, lay white eggs, but many of the parasitic species lay colored eggs to match those of their passerine hosts. In some cases female cuckoo can mimic the color and pattern of eggs of some selected host nets [37]. This feature minimizes the probability of eggs being thrown away from the nest and causes an increment in productivity of cuckoos further [38]. Non-parasitic cuckoos leave the nest before they can fly, and some new world species have the shortest incubation periods among birds. The cuckoo breeding process is based on the current position of cuckoo and probability of better next position after a selected random walk with a number of chosen random step sizes. This random walk plays a major role in the exploration, exploitation, intensification and diversification of the breeding process [39]. In general this foraging of random walk and step size follows a probability distribution function. The probability distribution functions like Gaussian distribution, normal distribution, and Levy distribution [40]. In cuckoo search, random walk follows levy flight and step size follows levy distribution function as given in Eq. (21). Levy flight is a random walk whose step follows the levy distribution function. In huge search space levy flight random walk is better than Brownian walk because of its nonlinear sharp variation of parameters. The direction of walk follows the uniform distribution function and steps of walk follow Mantegna's algorithm which gives both positive and negative numbers. The Levy distribution function is

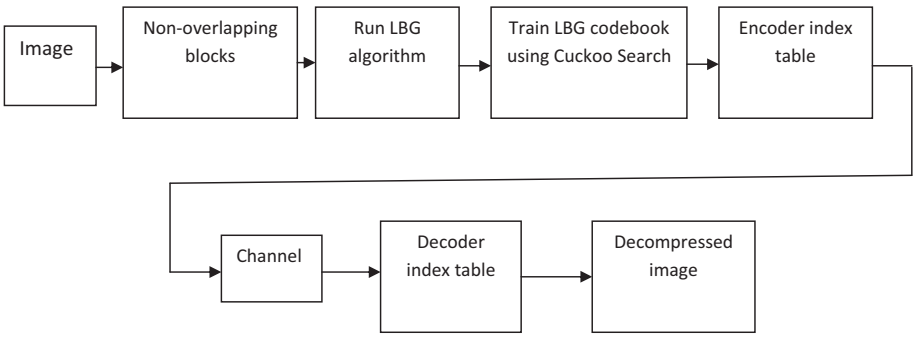


Figure 3 Block diagram of the proposed methodology.

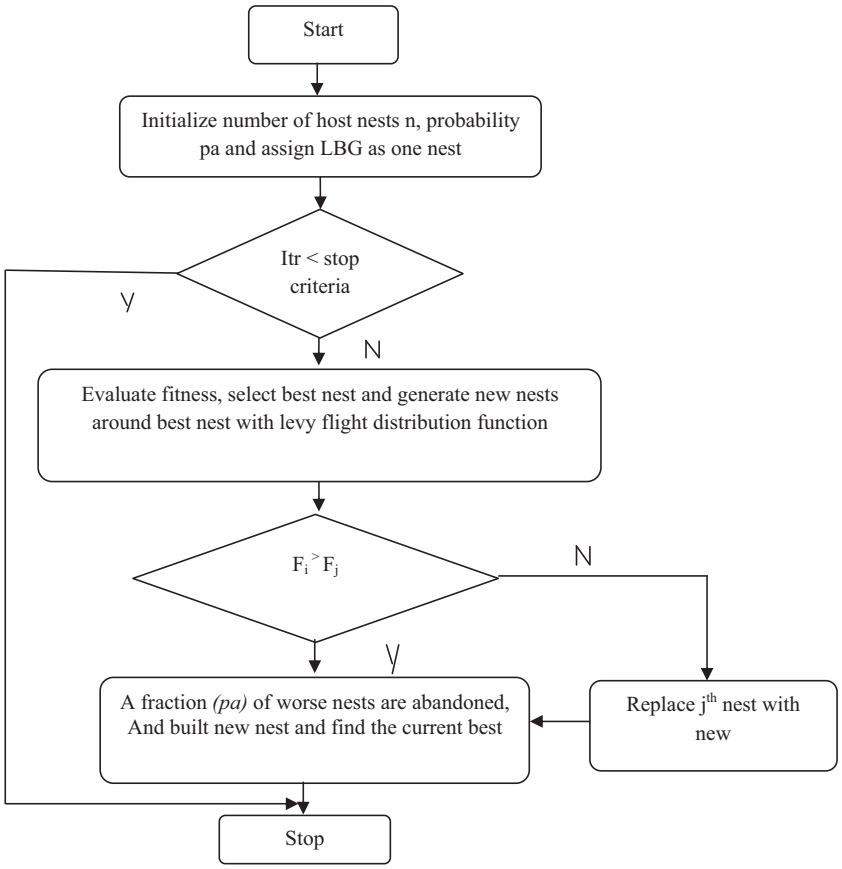


Figure 4 Flowchart of CS-LBG algorithm.

$$L(s,\gamma,\mu) = \begin{matrix} \sqrt{\frac{\gamma}{2\pi}}\exp\left[-\frac{\gamma}{2(s-\mu)}\right]\frac{1}{(s-\mu)^{3/2}} & 0<\mu<s<\infty \\ 0 & \text{otherwise} \end{matrix} \quad (15)$$

where $\mu > 0$ is a minimum step and γ is the scale parameter. If $s \to \infty$ then Eq. (15) becomes

$$L(s,\gamma,\mu) \approx \sqrt{\frac{\gamma}{2\pi}}\frac{1}{(s)^{3/2}} \quad (16)$$

In cuckoo search, the algorithm for generation of random walk step is based mostly on Mantegna's algorithm. According to Mantegna's algorithm the step size of random walk of the cuckoo is given by Eq. (17)

$$\text{Step of random walk} = \frac{\mu}{(v)^{1/\beta}} \quad (17)$$

where μ and v are drawn from normal distribution or gaussian distribution is given in Eq. (18) with $\beta = 2$

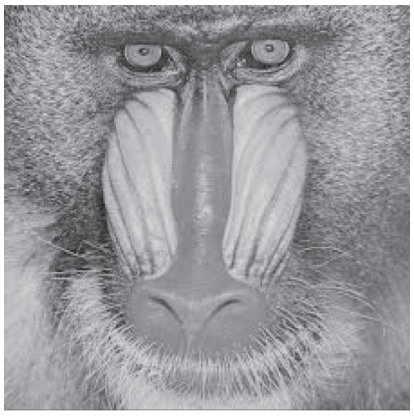
$$L(s) = \frac{1}{\pi}\int_0^\infty \cos(\tau s)e^{-\tau\alpha^\beta}d\tau \quad (18)$$

From above equation

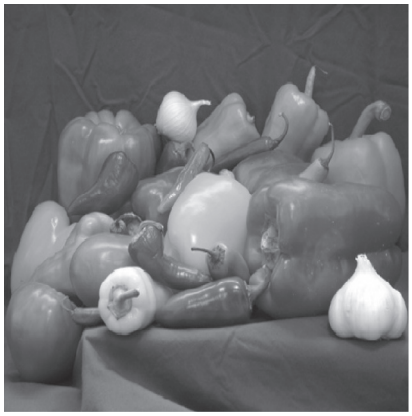
$$\mu \approx \mathbf{N}(0,\sigma_\mu^2) \quad v \approx N(0,\sigma_v^2) \quad (19)$$

where $N()$ normal distribution function is given in Eq. (20)

(a) LENA image



(b) BABOON image



(c) PEPPERS image

(d) BARB image

(e) GOLDHILL image

Figure 5 The five test images: (a) LENA, (b) BABOON, (c) PEPPERS, (d) BARB and (e) GOLDHILL.

$$N(\mu,\sigma^2)=\frac{1}{\sigma\sqrt{2\pi}}\exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right] \quad -\infty<x<\infty \tag{20}$$

where

$$\sigma\mu=\left\{\frac{\Gamma(1+\beta)\sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left[\frac{1+\beta}{2}\right]\beta 2^{(\beta-1)/2}}\right\}^{\frac{1}{\beta}} \quad \text{and} \quad \sigma_v=1 \tag{21}$$

where gamma function (Γ) is given in Eq. (22)

$$\Gamma(\beta)=\int_0^\infty e^{-t}t^{\beta-1}dt \tag{22}$$

The block diagram of vector quantization using a cuckoo search algorithm is shown in Fig. 3. An image to be vector quantized is divided into immediate and non-overlapping blocks. These non-overlapping blocks are vector quantized with an LBG algorithm. The generated codebook of LBG algorithm is trained with the cuckoo search algorithm that satisfies the global convergence requirements and guarantees the

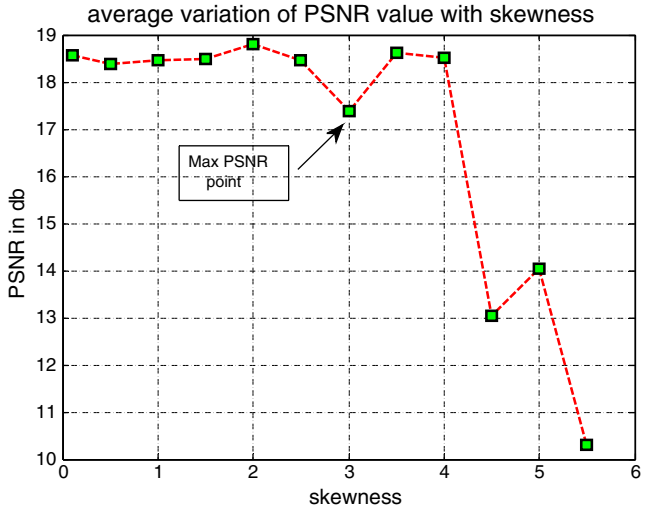


Figure 6 Average PSNR of LENA image being Performed 5 times for selection of skewness parameter.

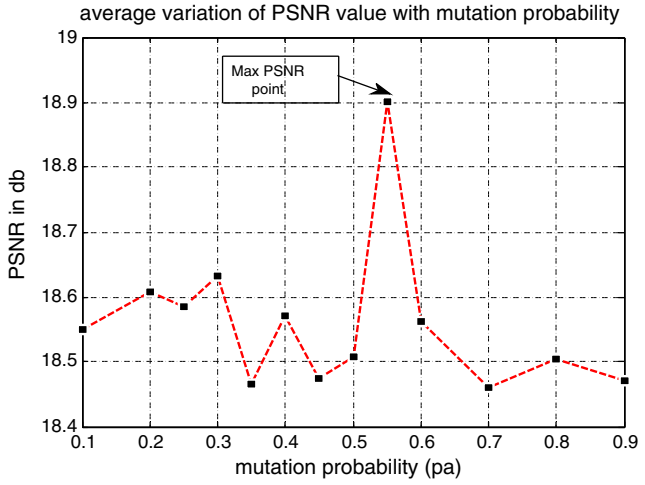


Figure 7 Average PSNR of LENA image being Performed 5 times for selection of mutation probability.

Table 1 The parameters used in the FA-LBG algorithm.

Parameter	Explanation	Value
n	Population size	30
iter	Iterations	20
α	Alpha	0.01
β_0	Beta minimum	1
γ	Gamma	1

Table 2 The parameters used in the CS-LBG algorithm.

Parameter	Explanation	Value
n	Population size	30
iter	Iterations	20
Pa	Mutation probability	0.55
β	Beta	2

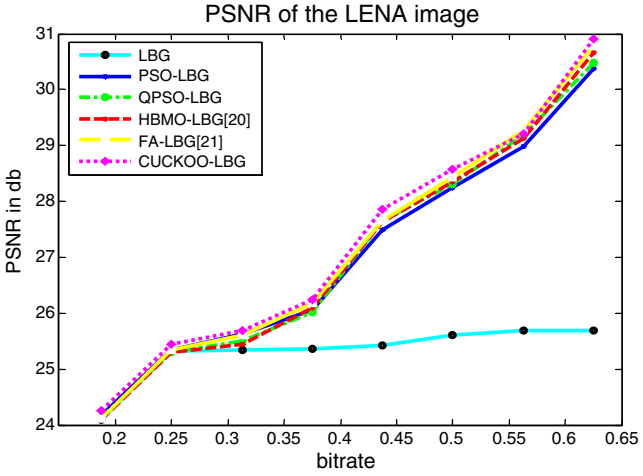


Figure 8 The average PSNR of six vector quantization methods for LENA image.

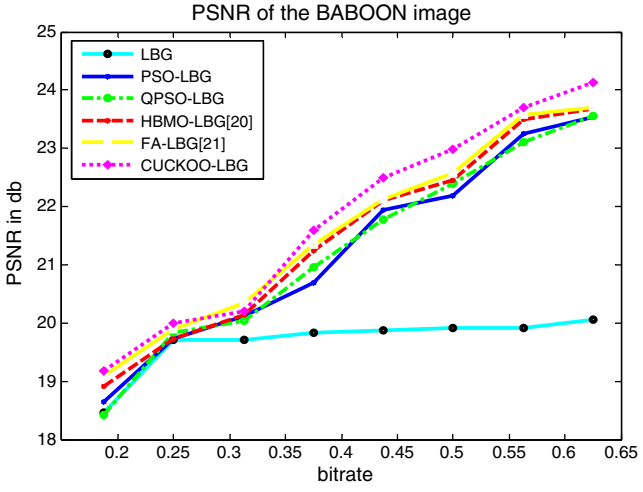


Figure 9 The average PSNR of six vector quantization methods for BABOON image.

global convergence properties. Furthermore, cuckoo search is able to search for a local codebook and global codebook with the help of control parameter called mutation probability (P_a). Mutation probability of 0.25 gives local codebook with 25% of convergence time and global codebook takes 75% of convergence time. Assign each non-overlapping block of the input image to one of the nearest codeword of trained codebook and its corresponding index number forms index table. These index numbers are transmitted over the channel and retrieved back with the help of the decoder at the receiver. All the decoded index numbers and the corresponding codewords are rearranged in a manner that the decompressed image size is the same as that of the input image.

Cuckoo search algorithm works with following three idealized rules: Each cuckoo lays one egg at a time, and dumps it in a randomly chosen nest; the best nest with high quality of eggs (solutions) will carry over to the next generations; the number of available host nests is fixed, and a host can discover an alien egg with a probability $P_a \in [0, 1]$. In this case, the host bird can either throw the egg away or abandon the nest so as to build a

Please cite this article in press as: Chiranjeevi K, Jena UR, Image compression based on vector quantization using cuckoo search optimization technique, Ain Shams Eng J (2016), http://dx.doi.org/10.1016/j.asej.2016.09.009

completely new nest in a new location. Here each cuckoo nest is assumed as codebook. The flowchart of the cuckoo search algorithm is summarized in Fig. 4. The detailed algorithm for vector quantization is as follows:

Step 1: (Initialization of parameters and solutions): Initialize number of host nests with each nest containing a single egg, mutation probability (P_a) and a tolerance. Run the LBG algorithm and assign its outcome as one of the nest/egg and rest nests randomly.
Step 2: (selection of the current best solution): Calculate the fitness of all nests using Eq. (1) and select maximum fitness nest as current best nest $nest_{best}$.
Step 3: (Generate new solutions with Mantegna's algorithm): New cuckoo nests ($nest_{new}$) are generated which are around current best nest with random walk (Levy flight). This random walk follows Levy distribution function which obeys Mantegna's algorithm. New nest is given as

$$nest_{new} = nest_{old} + \alpha \otimes Le'vy(\lambda) \tag{23}$$

where α is step size usually equal to one and Levy (λ) is Levy distribution function is given in Eqs. (15), (16), (17) and (21). For the sake of simplicity Eq. (23) is modified as

$$nest_{new} = nest_{old} + step \times (nest_{best} - nest) \tag{24}$$

where step is a random walk follows Levy distribution function, Eq. (17)
Step 4: (discard worst nets and replace with new nests): If the generated random number (K) is greater than mutation probability (P_a) then replace worse nests with new nests by keeping the best nest unchanged. New nests are generated by a random walk and random step size is given as

$$nest_{new} = nest_{old} + (K \times stepsize) \tag{25}$$

where

$$stepsize = r \times (nest_{rand} - nest_{rand}) \quad (r \text{ is random number}) \tag{26}$$

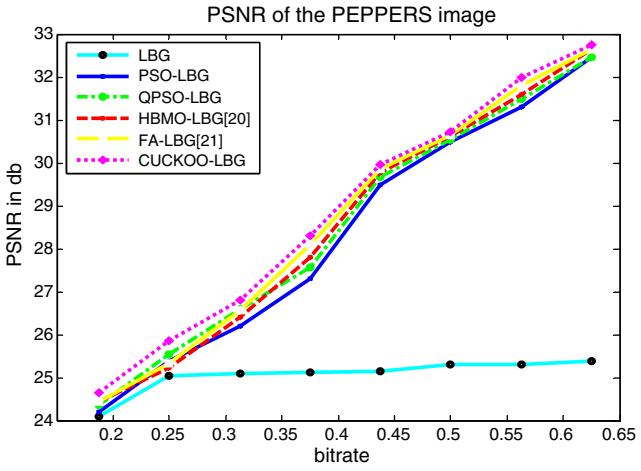


Figure 10 The average PSNR of six vector quantization methods for PEPPER image.

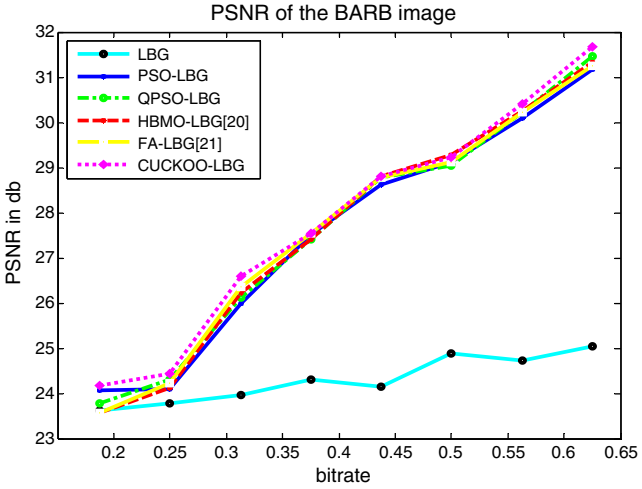


Figure 11 The average PSNR of six vector quantization methods for BARB image.

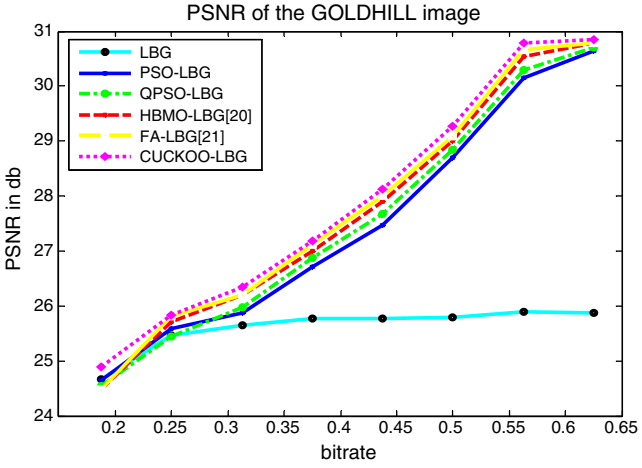


Figure 12 The average PSNR of six vector quantization methods for GOLDHILL image.

Step 5: Rank the nests based on fitness function and select the best nest.
Step 6: Repeat step 2 to step 4 until termination criteria.

4. Simulation results and discussion

The typical experiments for evaluating the methods used for codebook design are the grayscale image coding. Five different images, "Lena", "Baboon", "Pepper", "Barb" and "Goldhill" are chosen for comparison of cuckoo search algorithm with other algorithms and are shown in Fig. 5a–e respectively. All the images are grayscale images of size 512 × 512 pixels [21]. Among all the images pepper is ".png" format and remaining images are ".jpg" format. All the images are compressed with CS-LBG, FA-LBG, HBMO-LBG, QPSO-LBG, PSO-LBG and LBG. As discussed in Section 2.1, the image to be compressed is subdivided into non-overlapping images of size 4 × 4 pixels. Each subdivided images called blocks are treated as a training vector of (4 × 4) 16 dimensions. So there are

Table 3 The average computation time of the test images by using the six different algorithms with the bit rate = 0.1875 and codebook size = 8.

Image	Average computation time (s)					
	LBG	PSO-LBG	QPSO-LBG	HBMO-LBG	FF-LBG	CS-LBG
LENA	3.371542	254.357919	261.299374	890.254347	877.123889	977.200938
PEPPER	3.416869	247.181509	252.652972	676.344334	660.141175	1019.697575
BABOON	4.313500	322.996476	326.493443	723.897865	705.210035	1411.392008
GOLDHILL	3.622867	247.211833	346.986628	681.978545	661.281546	1069.057758
BARB	3.843362	257.520535	268.403187	654.976675	631.435366	1680.334183
Average	3.713628	265.8536544	291.1671208	725.4903532	707.0384022	1231.536492

Table 4 The average computation time of the test images by using the six different algorithms with the bit rate = 0.25 and codebook size = 16.

Image	Average computation time (s)					
	LBG	PSO-LBG	QPSO-LBG	HBMO-LBG	FF-LBG	CS-LBG
LENA	3.405184	252.001592	267.208254	528.995495	507.183026	1006.583897
PEPPER	4.575627	250.411978	253.430517	567.656548	534.304628	1708.635425
BABOON	4.537825	321.669194	333.844743	952.324245	943.362724	1455.553349
GOLDHILL	4.907262	318.004351	376.746590	589.097844	574.986090	1261.561267
BARB	4.391757	264.414665	312.586584	745.876776	737.332125	1337.834255
Average	4.363531	281.300356	308.7633376	676.7901816	659.4337186	1354.033639

Table 5 The average computation time of the test images by using the six different algorithms with the bit rate = 0.3125 and codebook size = 32.

Image	Average computation time (s)					
	LBG	PSO-LBG	QPSO-LBG	HBMO-LBG	FF-LBG	CS-LBG
LENA	5.262231	306.532039	318.314149	761.346655	710.810851	1316.946133
PEPPER	6.241595	338.618858	272.908139	571.875346	594.783631	1090.371821
BABOON	5.171656	272.346143	289.707195	726.638634	723.566566	1579.699665
GOLDHILL	4.481844	277.087057	313.025201	779.098764	755.606849	1525.981190
BARB	6.675448	281.853149	315.750745	896.897654	877.698960	1346.776353
Average	5.5665548	295.287449	301.941085	747.171410	732.493371	1371.955032

Table 6 The average computation time of the test images by using the six different algorithms with the bit rate = 0.3750 and codebook size = 64.

Image	Average computation time (s)					
	LBG	PSO-LBG	QPSO-LBG	HBMO-LBG	FF-LBG	CS-LBG
LENA	4.958598	311.440476	320.119462	745.345347	711.452695	1429.493919
PEPPER	5.768537	305.034406	305.952327	636.967533	633.629908	1531.719047
BABOON	6.728556	314.850105	323.926381	775.232423	768.088085	1507.091245
GOLDHILL	8.795603	382.040368	391.577189	968.356788	934.453184	1907.092025
BARB	11.21273	308.216570	312.251758	874.778777	846.853926	1369.893529
Average	7.4928048	324.316385	330.7654234	800.1361736	778.8955596	1549.057953

16,384 $\left(\frac{512}{4} \times \frac{512}{4}\right)$ input vectors to be encoded using a codebook which is designed by any one of the algorithms.

The parameters used for comparison of proposed cuckoo search algorithm with others are bit rate/bits per pixel (bpp), Peak Signal to Noise Ratio (PSNR) and Mean Square Error (MSE) as given in Eqs. (27), (28) and (29) respectively. PSNR and fitness values are calculated for all the images with different codebook sizes of 8, 16, 32, 64, 128, 256, 512 and 1024

Please cite this article in press as: Chiranjeevi K, Jena UR, Image compression based on vector quantization using cuckoo search optimization technique, Ain Shams Eng J (2016), http://dx.doi.org/10.1016/j.asej.2016.09.009

Table 7 The average computation time of the test images by using the six different algorithms with the bit rate = 0.4375 and codebook size = 128.

Image	Average computation time (s)					
	LBG	PSO-LBG	QPSO-LBG	HBMO-LBG	FF-LBG	CS-LBG
LENA	11.888536	522.284262	534.054431	889.324333	866.079748	1609.658552
PEPPER	16.421532	507.584419	570.995410	957.734356	914.451402	1876.924076
BABOON	19.623416	405.582369	465.000805	964.673393	920.112334	2195.838932
GOLDHILL	15.216410	697.720161	718.694109	1180.36544	1122.441935	1457.215903
BARB	27.346822	521.941565	531.267813	1164.09897	1145.650131	2044.023465
Average	18.0993432	531.0225552	564.002513	1031.239298	993.7471132	1836.732186

Table 8 The average computation time of the test images by using the six different algorithms with the bit rate = 0. 50 and codebook size = 256.

Image	Average computation time (s)					
	LBG	PSO-LBG	QPSO-LBG	HBMO-LBG	FF-LBG	CS-LBG
LENA	20.913288	875.945256	894.622455	799.356456	789.138474	1597.380117
PEPPER	18.116559	750.584419	750.584419	997.987876	972.207816	1707.318446
BABOON	28.028078	594.620811	563.620811	1011.56545	1032.44629	2006.820215
GOLDHILL	29.701560	924.444311	556.342111	843.534555	827.919726	2932.646895
BARB	27.619608	683.999751	692.899838	840.246767	830.791279	2555.872356
Average	24.8758186	765.9189096	691.6139268	898.5382208	890.500717	2160.007606

Table 9 The average computation time of the test images by using the six different algorithms with the bit rate = 0.5625 and codebook size = 512.

Image	Average computation time (s)					
	LBG	PSO-LBG	QPSO-LBG	HBMO-LBG	FF-LBG	CS-LBG
LENA	56.316994	1156.908559	1196.316994	1790.444554	1716.80639	2638.437774
PEPPER	82.002117	1950.230780	1851.809192	1387.908655	1357.86469	2862.149227
BABOON	63.433322	2010.197209	2187.580293	2178.565466	2212.438038	3544.228924
GOLDHILL	77.850099	1291.175546	1332.753876	2116.445100	2126.344435	2776.046559
BARB	115.21079	1296.291040	1123.215648	1396.123567	1386.554184	2868.870087
Average	78.9626644	1540.960627	1538.335201	1773.897468	1760.001547	2937.946514

Table 10 The average computation time of the test images by using the six different algorithms with the bit rate = 0.625 and codebook size = 1024.

Image	Average computation time (s)					
	LBG	PSO-LBG	QPSO-LBG	HBMO-LBG	FF-LBG	CS-LBG
LENA	145.725844	3422.799272	3518.889707	4290.667555	4229.809396	8272.419424
PEPPER	140.346789	2262.336878	2558.192815	2773.786877	2723.738130	4555.170085
BABOON	156.576716	3376.804469	3386.370076	3914.987866	3848.840251	5637.187818
GOLDHILL	181.405248	2594.207355	2624.493640	2854.564565	2842.180938	4485.623620
BARB	211.115436	2847.841612	2826.772350	2254.343435	2221.664446	4511.959378
Average	167.0340066	2900.797917	2982.943718	3217.670064	3173.246632	5492.472065

codewords. The evaluation of data size of the compressed image for various codebook sizes of 8, 16, 32, 64, 128, 256, 512 and 1024 is carried out using bpp and quality of the reconstructed image is assessed by PSNR.

$$bpp = \frac{\log_2 N_c}{k} \tag{27}$$

where Nc is codebook size and k is the size of a block.

Please cite this article in press as: Chiranjeevi K, Jena UR, Image compression based on vector quantization using cuckoo search optimization technique, Ain Shams Eng J (2016), http://dx.doi.org/10.1016/j.asej.2016.09.009

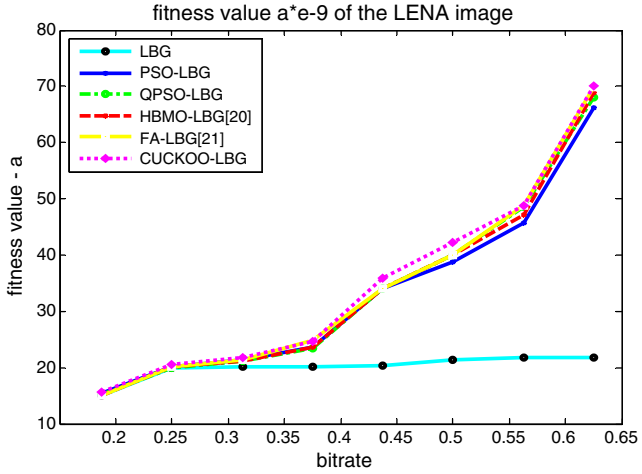


Figure 13 The average fitness values of six vector quantization methods for LENA image.

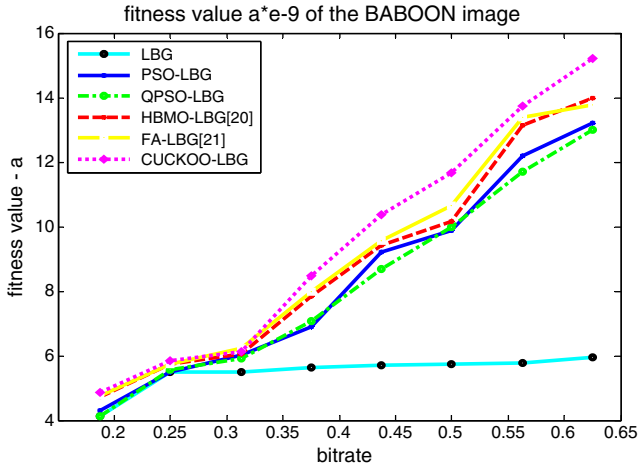


Figure 14 The average fitness values of six vector quantization methods for BABOON image.

$$PSNR = 10 \times 10 \log \left(\frac{255^2}{MSE} \right) (dB) \tag{28}$$

where MSE is given in Eq. (29)

$$MSE = \frac{1}{M \times N} \sum_{I}^{M} \sum_{J}^{N} \{ f(I,J) - \bar{f}(I,J) \}^2 \tag{29}$$

where $M \times N$ is the size of the image, and I and J represent the coordinate values of pixel position of both the original and decompressed images. In our experiment, we have taken $M = N$ a square image. $f(I,J)$ is the original image and $\bar{f}(I,J)$ is the reconstructed image.

The parameter values of the cuckoo search algorithm used for simulating all the images are chosen based on the occurrence of maximum average PSNR value of experiments performed (three times in our case). Skewness parameter (β) = 2 and the mutation probability (P_a) = 0.55 are selected for optimization of the codebook based on the occurrence of maximum PSNR value as shown in Figs. 6 and 7 respectively. The parameters used for simulating PSO-LBG, QPSO-LBG and HBMO-LBG are the same as those referred in paper

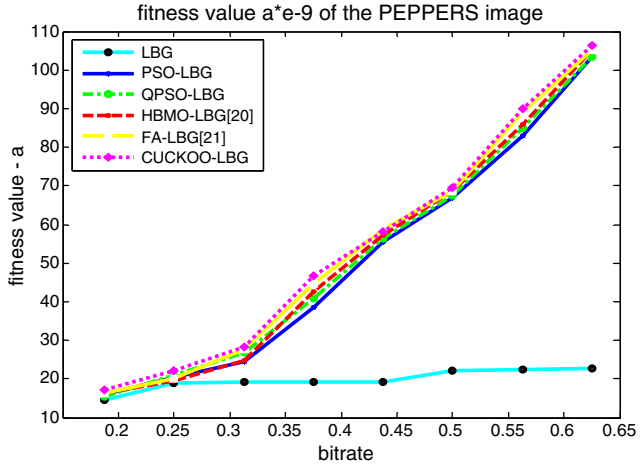


Figure 15 The average fitness values of six vector quantization methods for PEPPERS image.

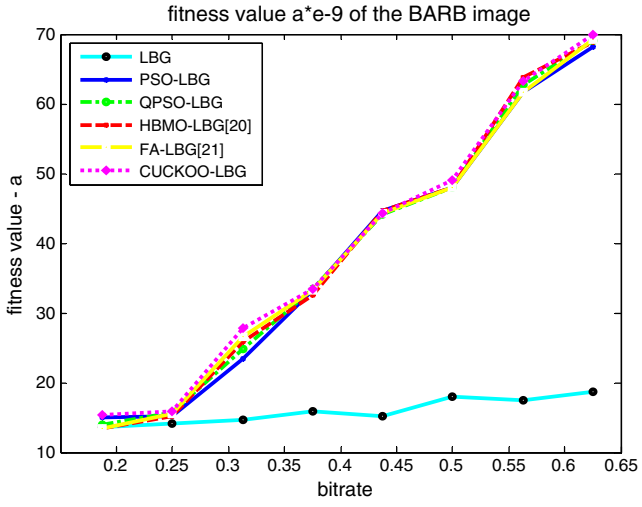


Figure 16 The average fitness values of six vector quantization methods for BARB image.

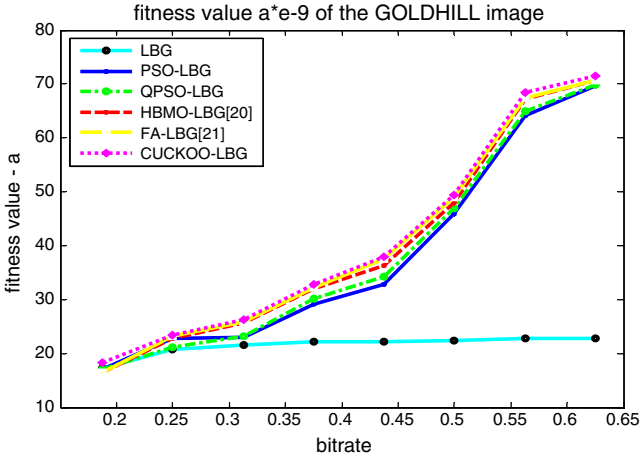


Figure 17 The average fitness values of six vector quantization methods for GOLDHILL image.

Figure 18 Decompressed BARB image of six VQ techniques with codebook size of 256 (a) LBG, (b) PSO-LBG, (c) QPSO-LBG, (d) HBMO-LBG, (e) FA-LBG, (f) CS-LBG.

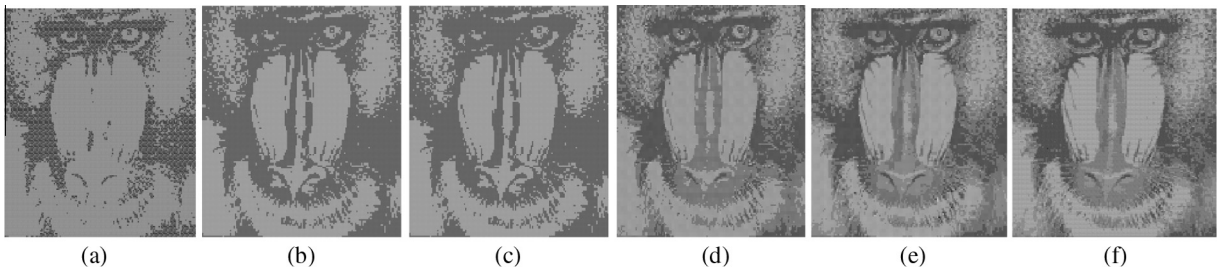


Figure 19 Decompressed BABOON image of six VQ techniques with codebook size of 256 (a) LBG, (b) PSO-LBG, (c) QPSO-LBG, (d) HBMO-LBG, (e) FA-LBG, (f) CS-LBG.

Figure 20 Decompressed GOLDHILL image of six VQ techniques with codebook size of 256 (a) LBG, (b) PSO-LBG, (c) QPSO-LBG, (d) HBMO-LBG, (e) FA-LBG, (f) CS-LBG.

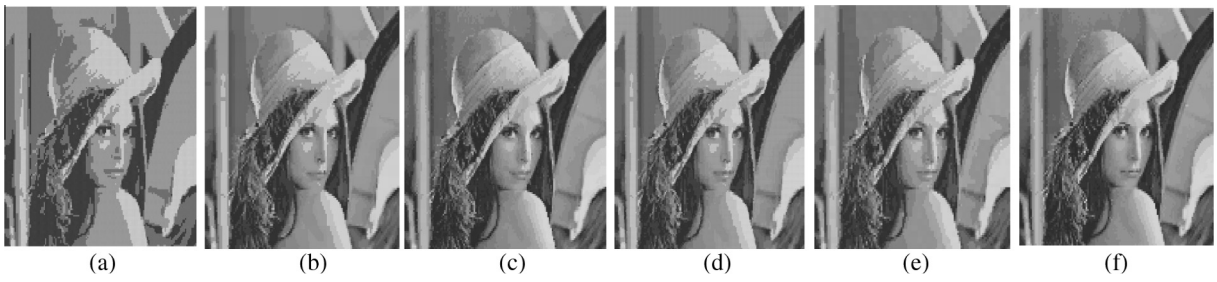


Figure 21 Decompressed LENA image of six VQ techniques with codebook size of 256 (a) LBG, (b) PSO-LBG, (c) QPSO-LBG, (d) HBMO-LBG, (e) FA-LBG, (f) CS-LBG.

[21]. The parameters used for simulating FA-LBG and CS-LBG are tabulated in Tables 1 and 2 respectively. To understand the performance of the proposed method, graphs showing the variation of average peak signal to noise ratio with respect to bit rate are plotted for each method. Figs. 8–12 show the average peak signal to noise ratio of different tested images against bit rate. Experimentally, it shows that the CS algorithm improves the PSNR values by around 0.2 dB at low bit rate and 0.3 dB at a higher bit rate. Experimentally, it is observed from the graphs that, for different codebook sizes, CS algorithm PSNR value is better than LBG, PSO-LBG, QPSO-LBG, HBMO-LBG and FA-LBG.

Please cite this article in press as: Chiranjeevi K, Jena UR, Image compression based on vector quantization using cuckoo search optimization technique, Ain Shams Eng J (2016), http://dx.doi.org/10.1016/j.asej.2016.09.009

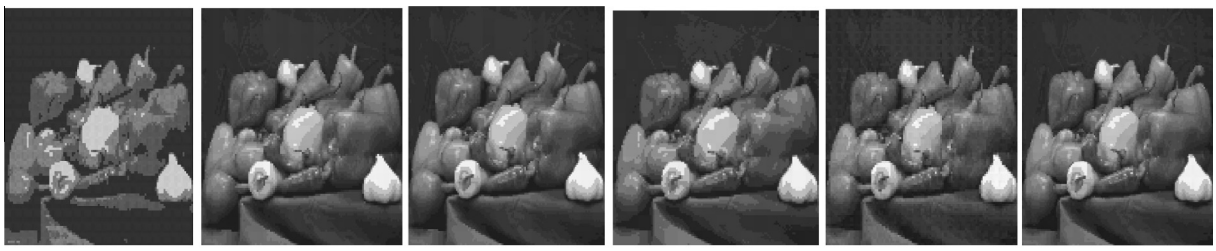


Figure 22 Decompressed PEPPER image of six VQ techniques with codebook size of 256 (a) LBG, (b) PSO-LBG, (c) QPSO-LBG, (d) HBMO-LBG, (e) FA-LBG, (f) CS-LBG.

The empirical simulation is performed on windows XP operating system with an Intel(R) Core(TM) i5-2540 and 2.60 GHz CPU with 2.94 GB RAM. Moreover, all the programs are written and compiled on MATLAB version 7.9.0 (R2009b). Tables 3–10 show the average computation time or convergence time of the different algorithms with different bit rates. Horng in his paper [20] simulated the five algorithms in 'C++6.0' with windows XP operating systems, taking 100 numbers of codebooks/solutions and 50 numbers of iterations. In our work, the six algorithms are simulated in MATLAB with 30 numbers of codebooks and 20 numbers of iterations. So there is some dissimilarity of average computational time between proposed CS-LBG and FA-LBG. In a similar way, Horng in his paper [21] simulated the five algorithms in MATLAB with 100 iterations. From observations of Tables 3–10, LBG algorithm computational time is significantly lower as compared to all other algorithms, but of lesser PSNR and bad reconstructed image quality. On average, the CS algorithm is around 1.425 times slower than the firefly algorithm and honey bee mate optimization algorithm. Results and analysis also imply that the convergence rate, to some extent, is not sensitive to the parameter P_a. CS algorithm is easy to implement because tuning of mutation probability is enough to design a global codebook, whereas for other methods many parameters are required to tune for global codebook design. The normal fitness values of the five experimented images using six vector quantization algorithms are plotted in Figs. 13–17. The investigations confirmed that the fitness of the five test images using the CS-LBG algorithm is higher than the LBG, PSO-LBG, QPSO-LBG, HBMO-LBG and FA-LBG algorithm. Figs. 18–22 show the decompressed/reconstructed images of five images obtained by six vector quantization methods with a codebook size of 256 and block size of 16. It is observed that the decompressed/reconstructed image quality of the CS-LBG algorithm is superior to the quality of reconstructed images of LBG, PSO-LBG, QPSO-LBG, HBMO-LBG and FA-LBG algorithms.

5. Conclusions

In this paper, a cuckoo search algorithm based vector quantization is proposed for image compression. The peak signal to noise ratio of vector quantization is maximized by employing CS algorithm. The algorithm has been investigated by varying all possible parameters of CS for efficient codebook design and efficient vector quantization of training vectors. Intensification and diversification of the algorithm are achieved with mutation probability and skewness parameter. Intensification intends to search around the current best solutions and to select the best solutions, while diversification makes sure that the algorithm can explore the search space more efficiently, often by randomization. It is observed that the peak signal to noise ratio and quality of the reconstructed image obtained with CS algorithm are superior to those obtained with LBG, PSO-LBG, QPSO-LBG, HBMO-LBG and FA-LBG. From the simulation results it is observed that CS-LBG is around 1.425 times slower in convergence as compared to HBMO-LBG and FA-LBG. Slower convergence is the major drawback of the proposed method and will be improved by modifications in the algorithm in future scope. However, the CS-LBG algorithm requires less parameters than PSO-LBG, QPSO-LBG, HBMO-LBG and FA-LBG.

References

[1] Taubman D, Marcellin MW. Jpeg-2000 image compression: fundamentals, standards and practice. Dordrecht: Kluwer Academic Publishers; 2002.
[2] Linde Y, Buzo A, Gray RM. An algorithm for vector quantize design. IEEE Trans Commun 1980;28:84–95.
[3] Patane G, Russo M. The enhanced LBG algorithm. Neural Networks 2002;14:1219–37.
[4] Jung Kyeong, Choong WL. Image compression using projection vector quantization with quadtree decomposition. Signal Process: Image Commun 1996;3:79–386.
[5] Abouali AH. Object-based VQ for image compression. Ain Shams Eng J 2015;6(1):211–6.
[6] Hu YC, Chang. Quadtree-semented image coding schemes using vector quantization and block truncation coding. Optim Eng 2000;2(39):464–71.
[7] Kazuya S, Sato S, Junji M, Yukinori S. Vector quantization of images with variable block size. Appl Soft Comput 2008;8:634–45.
[8] Dimitrios T, George ET, John T. Fuzzy vector quantization for image compression based on competitive agglomeration and a novel codeword migration strategy. Eng Appl Artif Intell 2012;25:1212–25.
[9] Dimitrios T, George ET, Antonios DN, Anastasios R. On the systematic development of fast fuzzy vector quantization for grayscale image compression. Neural Networks 2012;36:83–96.
[10] Xiaohui L, Jinchang R, Chunhui Z, Tong Q, Stephen M. Novel multivariate vector quantization for effective compression of hyperspectral imagery. Opt Commun 2014;332:192–200.
[11] Wang X, Meng J. A 2-D ECG compression algorithm based on wavelet transform and vector quantization. Digital Signal Process 2008;18:179–88.
[12] Rajpoot A, Hussain A, Saleem K, Qureshi Q. A novel image coding algorithm using ant colony system vector quantization. In: International workshop on systems, signals and image processing, Poznan, Poland.

[13] Tsaia CW, Tsengb SP, Yang CS, Chiang MC. PREACO: a fast ant colony optimization for codebook generation. Appl Soft Comput 2013;13:3008–20.
[14] Chen Q, Yang JG, Gou J. Image compression method by using improved P SO vector quantization. In: First international conference on neural computation (ICNC 2005). Lecture notes on computer science, vol. 3612. p. 490–5.
[15] Feng H, Chen C, Fun Y. Evolutionary fuzzy particle swarm optimization vector quantization learning scheme in image compression. Exp Syst Appl 2007;32:213–22.
[16] Wang Y, Feng XY, Huang YX, Zhou WG, Liang YC. A novel quantum swarm evolutionary algorithm and its applications. Neurocomputing 2007;70:633–40.
[17] Chang C, Li YC, Yeh J. Fast codebook search algorithms based on tree-structured vector quantization. Pattern Recogn Lett 2006;27:1077–86.
[18] Yu-Chen H, Bing H, Chih CT. Fast VQ codebook search for gray scale image coding. Image Vis Comput 2008;26:657–66.
[19] Sanyal N, Chatterjee A, Munshi. Modified bacterial foraging optimization technique for vector quantization-based image compression. Computational intelligence in image processing. Springer; 2013, p. 131–52.
[20] Horng M, Jiang T. Image vector quantization algorithm via honey bee mating optimization. Exp Syst Appl 2011;38:1382–92.
[21] Horng M. Vector quantization using the firefly algorithm for image compression. Exp Syst Appl 2012;39:1078–91.
[22] Chiranjeevi K, Jena UR, Murali B, Jeevan K. Modified firefly algorithm (MFA) based vector quantization for image compression. In: Computational Intelligence in Data Mining (ICCIDM), Advance in intelligent systems and computing, vol. 2; 2015. p. 373–81.
[23] Chiranjeevi K, Jena UR. Fast vector quantization using bat algorithm for image compression. Int J Eng Sci Technol 2016;19:769–81.
[24] Lakshmia M, Senthilkumar J, Suresh Y. Visually lossless compression for Bayer color filter array using optimized Vector Quantization. Appl Soft Comput 2016;46:1030–42.
[25] Kennedy J, Eberhart RC. A new optimizer using particle swarm theory. In: Proceedings of sixth international symposium on micro machine and human science, Nagoya, Japan. p. 39–43.
[26] Chen Q, Yang JG, Gou J. Image compression method using improved PSO vector quantization. In: First international conference on neural computation (ICNC 2005). Lecture notes on computer science, vol. 3612. p. 490–5.
[27] Baykasoglu A, Ozbakor L, Tapkan P. Artificial bee colony algorithm and its application to generalized assignment problem. In: Chan FTS, Tiwari MK, editors. Swarm intelligence. I-Tech Education and Publishing; 2007. p. 113–44.
[28] Karaboga D, Basturk B. On the performance of Artificial Bee Colony (ABC) algorithm. Appl Soft Comput 2008;8:687–97.
[29] Afshar A, Bozog OH, Marino MA, Adams BJ. Honey-Bee Mating Optimization (HBMO) algorithm for optimal reservoir operation. J Frank Inst 2007;344:452–62.
[30] Yang XS. Nature-inspired metaheuristic algorithms. Luniver Press; 2008.
[31] Yang XS. Firefly algorithms for multimodal optimization in stochastic algorithms: foundation and applications. Lect Notes Comput Sci 2009;5792:169–78.
[32] Qinghai B. Analysis of particle swarm optimization algorithm. Comput Inform Sci 2010;3:180–4.
[33] Surafel LT, Hong C. Modified firefly algorithm. J Appl Math 2012;12:120–32.
[34] Yang XS, Deb S. Cuckoo search via levey flights. In: Proceedings of the World congress on nature and biologically inspired computing, vol. 4; 2009. p. 210–14.
[35] Yang X, Sand DS. Engineering optimization by cuckoo search. Int J Math Model Numer Optim 2010;4(1):330–43.
[36] Chakraverty S, Kumar A. Design optimization for reliable embedded system using cuckoo search. In: International conference on electronics computer technology, vol. 1; 2011. p. 264–8.
[37] Valian E, Mohanna S, Tavakoli S. Improved cuckoo search algorithm global optimization. Int J Commun Inform Technol 2011;1:3–44.
[38] Payne RB, Sorenson MD, Klitz K. The cuckoos. Oxford University Press; 2005.
[39] Blum C, Roli A. Metaheuristics in combinatorial optimization: overview and conceptual comparison. J Comput Surv ACM Digit Lib 2003;35:268–308.
[40] Brown C, Liebovitch LS, Glendon R. Levy flights in Dobe Ju/'hoansi foraging patterns. Human Ecol 2007;35:129–38.

Karri Chiranjeevi received B.Tech. in 2003 from JNTU Hyderabad, Andhra Pradesh, India, M.tech. in 2003 from JNTU Kaikinada, Andhra Pradesh, India, and doing Ph.D. in Veer Surendra Sai University of Technology, Burla, Odisha, India. He is working as Assistant Professor in the department of Electronics and Communication Engineering in GMR institute of Technology, Rajam, Srikakula, India. His research interests include Image processing, Soft Computing Techniques and Communication Engineering. He is having an experience of teaching more than 11 years and also having national and international journals and conferences.

U.R. Jena Received B.SC. Engg. in 1983 from Sambalpur University, Odisha, India, M.tech. in 1997 from I.I.T. Kharagpur, India, and Ph. D. in 2003 from Jadavpur university, Kolkata, India. He is working as Professor in the department of Electronics and Telecommunication Engineering, Veer Surendra Sai University of Technology, Burla, Odisha, India. His research interests include computer vision, Image processing and soft computing. He is having an experience of teaching more than 25 years and also having national and international journals and conferences.

Please cite this article in press as: Chiranjeevi K, Jena UR, Image compression based on vector quantization using cuckoo search optimization technique, Ain Shams Eng J (2016), http://dx.doi.org/10.1016/j.asej.2016.09.009