PORT –SAID ENGINEERING RESEARCH JOURNAL
Faculty of Engineering –port said university
Volume 19 No. 1 March2015 pp:129: 136

A Hybrid Color Image Quantization Algorithm Based on K-means and Harmony Search Algorithms

Rehab F. Abdel-Kader [1], *Mohamed S.Yasein* [2] *and Asmaa Khaled* [3]

ABSTRACT

The importance of color image quantization is reduce the number of different colors in an image with minimum distortion so, it is one of the most important operations in the computer graphics and image processing. In this paper, a new algorithm for color image quantization based on a stand-alone harmony search (HS) algorithm is proposed as a first new algorithm. The second algorithm is a hybrid algorithm of K-means and HS. This algorithm is based on a clustering method. Clustering method is one of the most commonly used methods in color image quantization. Investigations results on some of the most commonly used test images in the quantization literature the experiences obtained with these proposed algorithms give high quality images compared with other methods.

KEYWORDS: Color quantization, Harmony search, K-means.

1. INTRODUCTION

Color quantization (CQ) is a technique in image processing used to reduce the number of colors in true color images and represented it by the color palette using only a small number of colors (usually between 8 and 256) [1]. Because the true color images contain thousands of different colors and thus the display, storage and transfer of the image are facing several problems, for this reason it is important to use a color image quantization. This color reduction process is essential in many applications such as video conferencing and image storage and transmission through limited bandwidth channels. Moreover, CQ is considered an efficient lossy compression technique which simplifies the image feature space and makes it more appropriate for image recognition and retrieval tasks [2, 3].

Color quantization process composed basically of two phases: palette design and pixel mapping. In palette design, a palette is created by choosing the small group of colors that are representing the colors of the original image (8-256 colors) [4]. In pixel mapping, assign each pixel in original image to one of the palette colors.

Most CQ methods tend to create an optimal color palette. The quality of the color palette depends mainly on the error between the original image and the reconstructed image which is usually characterized by a mean square error measure.

There are two kinds of methods for creating the color palette: image-independent methods and image-dependent methods.

Image-independent methods determine a generic palette without considering any specific image contents. On the other hand, image-dependent methods generate palettes based on the color distribution in the input images. The image-independent method often produces poor results because it does not take into account individual image characteristics although it is fast. Therefore, most of the recent work on color quantization relies on image-dependent methods which strive to attain a better balance between the computational complexity and the visual quality of the resultant image. Numerous image-dependent quantization methods have been reported and studied. These methods can be broadly categorized into two categories: pre-clustering (splitting) methods and post-clustering methods [2].

Well-known splitting methods include median-cut [5], variance-based method [6], center-cut [7], octree [8], binary splitting [9], greedy orthogonal bi-partitioning [10], and the radius weighted mean cut (RWM-cut) [11]. On the other hand, clustering algorithms adapted to CQ include maxmin [12], K-means(KM) [13], competitive learning [14], fuzzy c-means [15], and self-organizing maps [16].

The clustering technique is the foremost component in post clustering CQ. The KM algorithm is the most popular in this category.

Alternatively, researchers applied several stochastic nature-inspired optimization methods, such as Genetic Algorithm (GA) [17], Particle Swarm Optimization (PSO) [18], Bacteria Foraging Optimization (BFO) [19], Firefly algorithm [20], Honey Bee Optimization [21], and Differential Evolution [22].

In this paper a new algorithm to cluster colors for CQ based on the Harmony search (HS) algorithm is proposed. Two variants of the algorithm are examined. The first is based on a stand-alone HS algorithm while the second is a hybrid algorithm of KM and HS. To study the performance of the proposed method, different

[1]*Assoc. Prof, Electrical Engineering Department, computer and control Section, Faculty of Engineering, Port Said University, Port Said, Egypt ,E-mail: rehabfarouk @eng.psu.edu*
[2]*Lecturer, Electrical Engineering Department, computer and control Section, Faculty of Engineering, Port Said University, Port Said, Egypt , E-mail: myasein @eng.psu.edu*
[3]*Demonstrator, Electrical Engineering Department, computer and control Section, Faculty of Engineering, Port Said University, Port Said, Egypt, E-mail: eng_asmaakhaled @eng.psu.edu*

images from The USC-SIPI Image Database were tested as in literature [23].

The performance of the proposed algorithms has been evaluated by calculating the mean-squared error (MSE) and Peak Signal-to-Noise Ratio (PSNR) for the test images.

This paper is given an organized as follows: Section 2 describes K-means algorithm based on color quantization (KM-CQ) while Section 3 explains Harmony search algorithm (HS). Section 4 presents the implementation details of the two proposed algorithms: HS Color quantization (HS-CQ) and hybrid KM and HS(KMHS-CQ). The investigation results validating the efficiency of the proposed color quantization framework are presented in Section 5. Finally, Section 6 is a conclusion of this work.

2. K-MEANS COLOR QUANTIZATION(KM-CQ)

KM algorithm is an iterative clustering algorithm widely used in unsupervised learning due to its ease of implementation and convergence speed [13]. To perform clustering for CQ, colors in the original image are grouped into a small predefined number of clusters. A representative for each cluster is identified (cluster center). Then each color in original image is mapped to the nearest cluster center. The clustering for CQ problem can be mathematically formulated as an optimization problem that locates the optimal centroids of the clusters. Given an input image I with N color pixels, $I=[x_1, x_2,\ldots\ldots,x_N] \in R^D$, and the desired number of clusters K (for a color palette of K colors) where K< N.

The objective of KM-CQ is to partition I into K mutually exclusive clusters. As the clusters are disjoint, each pixel $x_i \in I$ will be assigned to exactly one cluster whose centroid C_k is closest to x_i. Each pixel $x_i \in I$ is then replaced by one of the K colors in the color palette. Consequently a new image I', called the quantized image of I is constructed. The objective of the algorithm is to find the optimal cluster configuration such that the total sum of distances between all the pixels and their corresponding cluster centroids is minimized.The total distance is typically computed by the Mean Squared Error (MSE) which is defined as follows:

$$\boldsymbol{MSE} = \sum_{\boldsymbol{k=1}}^{\boldsymbol{K}} \sum_{\boldsymbol{x_i \in a_k}} \|\boldsymbol{x_{i-}C_k}\|^{\boldsymbol{2}}, \qquad (1)$$

where $\|\boldsymbol{x_{i-}C_k}\|^{\boldsymbol{2}}$ denotes the Euclidean distance norm and $\boldsymbol{C_k}$ is the center of the $\boldsymbol{k}^{\text{th}}$ cluster.

KM-CQ consists of two basic processes: pixel assignment and centroid updating. By iteratively repeating these two processes, the required color palette is generated. KM-CQ can be summarized as follows:

Step 1: Generate the initial color palette (CP) of K colors by random selection from the set of image pixels.

Step 2: For each color pixel x_i, assign it to the closest cluster in the palette color such that the Euclidean distance between x_i and the cluster center c_k is minimal.

Step 3: For each cluster in *CP*, generate the new cluster center triplet c_k by a veraging all the pixels that have been assigned to the cluster. $\boldsymbol{c_k}$ is calculated as:

$$\boldsymbol{c_k} = \left[\frac{\sum_{i\in c_k}^{num_k} \boldsymbol{R_i^k}}{\boldsymbol{num_k}}, \frac{\sum_{i\in c_k}^{num_k} \boldsymbol{G_i^k}}{\boldsymbol{num_k}}, \frac{\sum_{i\in c_k}^{num_k} \boldsymbol{B_i^k}}{\boldsymbol{num_k}}\right], \qquad (2)$$

where num_k is the number of members in cluster $\boldsymbol{k}$, and $\boldsymbol{R_i^k}, \boldsymbol{G_i^k}, \boldsymbol{B_i^k}$ are the red, green, and blue levels of pixel p_i member of cluster $\boldsymbol{c_k}$.

Step 4: Repeat steps 2 and 3 until convergence (when no further change in the assignment of the pixels to clustersis achieved).

Step 5: Output the final palette of *K* colors.

3. HARMONY SEARCH ALGORITHM(HS)

HS is a population-based meta-heuristic first introduced by Geem et al. in 2001[24] for solving optimization problems. HS is inspired by the improvisation process of skilled musicians where the musicians try to improve their musical notes by combining the pitches of their instruments together to achieve a perfect harmony. To choose a decision variable in HS it follows one of three rules as in literature [24].

1. Choose a value from the Harmony memory (HM): defined as "consideration of the memory."
2. Choose a value adjacent to a value in HM: defined as "pitch adjustment."
3. Choose a random value from the range of possible values defined as "randomization".

HS is guided by four parameters:

1. Harmony memory size (*HMS*) (number of solution vectors in the solution of harmony memory);
2. Harmony memory considering rate (*HMCR*), where *HMCR* $\in$[0, 1];
3. Pitch adjusting rate (*PAR*), where *PAR* $\in$[0, 1].
4. Stopping Criteria (number of improvisations (*NI*)).

The main steps of HS algorithm are described as follows:

Step1: Initialize the HS parameters HMCR, PAR, BW, NI.

Step 2: Generate the initial population in HM.

Step 3: Improvise a new harmony from HM.

Step4: If the New Harmony is better than worst harmony in HM, then the worst harmony is replaced by the new harmony. Otherwise the new harmony is disregarded

Step 5: If stopping criteria is not satisfied, go to *Step 3*.

HS manages several vectors in a parallel manner similar to GA and PSO. However, HS possess some advantages compared to existing heuristics such as: HS makes the new vector after considering all existing vectors as opposed to considering only two existing (parents) as in GA, and HS does not require the initialization of the decision variables. These features help HS in increasing the flexibility of the search process and in finding better solutions.

4. PROPOSED ALGORITHMS

4.1. Preprocessing

In this stage, a preprocessing color compression scheme is applied to the original color image to reduce the number of distinct colors. Similar colors in the image pixel space are merged into the same color sphere. This is the concept of Histon, introduced by Mohabey and Ray for the conception of color information [25]. The main objective is exploiting the color redundancy in the original images by reducing the amount of data before the clustering phase.

Image histograms represent the pixel distribution in terms of intensity values of the color component. The histogram for each of the R, G, and B color components can be calculated as follows:

$$\boldsymbol{h_i(g)} = \sum_{\boldsymbol{m=1}}^{\boldsymbol{M}} \sum_{\boldsymbol{n=1}}^{\boldsymbol{N}} \delta(\boldsymbol{I(m,n,i)} - \boldsymbol{g}) \qquad \boldsymbol{for\ 0 \leq g \leq L-1}, \tag{3}$$

where $i=\{R, G, B\}$ and $\delta(\,)$ is the unit impulse function and L is the total number of intensity levels in each component. The value of $h_i(g)$ is the total number of image pixels having intensity g in image I. For every intensity value on the base histogram, the number of pixels encapsulated in the same color sphere is evaluated. This count is then added to the value of the histogram at that particular intensity value. The distance in the RGB color space is calculated by the Euclidean distance between and two pixels p_1 and p_2 is given as:

$$\boldsymbol{d(p_1,p_2)} = \sqrt{(\boldsymbol{R_1}-\boldsymbol{R_2})^2 + (\boldsymbol{G_1}-\boldsymbol{G_2})^2(\boldsymbol{B_1}-\boldsymbol{B_2})^2}, \tag{4}$$

A pixel p_1 in the input image I falls in the sphere of another pixel p_2 of a similar color if the distance $d(p_1, p_2)$ is less than the predefined threshold θ.

The processing steps of the color compression schema are described as follows:

Step 1: For each currently decoded pixel p∈I, if p belong to the first column or the first row in I go to Step 4.

Step 2: For each encoded neighboring pixel of p, compute the Euclidian distance between p and its neighboring pixel.

Step 3: Assign p to the compressed color index of neighboring pixel if the calculated distance ≤ θ, then go to Step 1. Otherwise go to Step 2.

Step 4: Compute the distance between p and previously encoded pixels. If a distance ≤ θ is found then assign p to the compressed color of the closest pixel. Go to Step 1.

Step 5: Construct a new color sphere for p. GotoStep1.

4.2. HS-CQ

In this section we describe the implementation details of the HS-CQ.

Step1: Initialize the Parameters of the Problem and HS Algorithm.

In this step, the HS parameters (*HMS*, *HMCR*, *PAR*, and *NI* or the stopping criteria) are specified.

The solution matrix HM has a size of *HMS*, where each harmony memory vector (decision variable) represents one candidate solution. In CQ each solution represents a possible combination of the *K* clusters centers defining the colors in the CP. Each harmony vector $C = (C_1, C_2, \ldots, C_K) = (c_{11}, c_{12},\ldots, c_{1D}, c_{21}, c_{22},\ldots, c_{2D},\ldots, c_{K1}, c_{K2},\ldots, c_{KD})$, has a length of ($K * D$), where D is the dimension of the solution space. In the RGB domain ($D=3$) three values for red, green and blue levels $\in[0, 255]$ are used to describe each cluster center. For example *C = (25, 145, 67, 139, 77, 200, 214, 99, 157)*, represents a possible solution with 3 cluster centers, which are $C_1= (25, 145, 67)$, $C_2= (139, 77, 200)$, $C_3= (214, 99, 157)$.

Step 2: Initialize the Harmony Memory

The HM is the location where all the solution vectors are stored. The HM is similar to the genetic population in the GA or the swarm in PSO. The HM matrix has *HMS* rows where each row is a *(3*K)* vector representing a possible cluster center combination.

The HM is randomly initialized and arranged as follows:

$$\boldsymbol{HM} = \left[\begin{array}{ccccc|c} C_{11}^{1} & C_{12}^{1} & \cdots & C_{K(D-1)}^{1} & C_{KD}^{1} & f(C^{1}) \\ C_{1}^{2} & C_{2}^{2} & \cdots & C_{K(D-1)}^{2} & C_{KD}^{2} & f(C^{2}) \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ C_{1}^{HMS} & C_{2}^{HMS} & \cdots & C_{K(D-1)}^{HMS} & C_{KD}^{HMS} & f(C^{HMS}) \end{array}\right], \tag{5}$$

where $f(C^i)$ is the fitness function of the solution vector C^i measuring the within-cluster variance and quantified by the MSE given in (1). The solutions are randomly constructed and rearranged in a reversed order in HM, according to their objective function values such that $f(C^1) \leq f(C^2) \cdots \leq f(C^{HMS})$.

Step 3: Improvise of New Harmony

This is the most-important step in the HS algorithms. In this step, three rules are used to generate a new harmony vector: memory consideration, pitch adjustment, and random selection.

In the memory consideration rule, the new value of the any decision variable c_{ij}^{new} is selected from the historical values stored for c_{ij} in HM with a probability of *HMCR*. Other decision variables not selected from the HM are randomly selected from their possible range of values. This is referred to as random consideration with a probability of (*1-HMCR*). The random consideration factor is induced to randomize the search process in a way to increase the diversity of the solutions and increases the probability of escaping local optima. The two rules are described as:

$$\mathbf{c_{ij}^{new}} = \begin{cases} \mathbf{c_{ij}^{new}} \in \{\mathbf{c_{ij}^{1}}, \mathbf{c_{ij}^{2}}, \dots, \mathbf{c_{ij}^{HMS}}\}, \textbf{With probability HMCR} \\ \mathbf{min} + (\mathbf{max} - \mathbf{min}) * \mathbf{rand(0,1)}, \\ \qquad \textbf{With probability } (\mathbf{1 - HMCR}) \end{cases} \quad (\mathbf{6})$$

where *min=0* and *max=255* are the lower and upper bounds for c_{ij}, and *rand()* is a function that generates a random number $\in$[0, 1]. Furthermore, the search process is enhanced thru fine-tuning the decision variables in the new HM. Each decision vector is examined and pitch adjusted with a probability of *PAR* as follows:

$$\mathbf{c_{ij}^{new}} = \begin{cases} \mathbf{c_{ij}^{old}} \pm \mathbf{rand(0,1)} * \mathbf{BW}, & \textbf{With probability PAR} \\ \mathbf{c_{old}(j)}, & \textbf{With probability } (\mathbf{1 - PAR}) \end{cases} \quad (7)$$

where the bandwidth factor (*BW*) is an arbitrary distance used to enhance the performance of HS by controlling the degree of movements within the components of the harmony vector.

Step 4: Update the Harmony Memory
The fitness function is calculated for each new harmony vector. If the new harmony has a better fitness value than that of the worst harmony in HM, then the new harmony will be included in the updated HM and the worst harmony will be excluded. Otherwise the new harmony will be ignored.

Step 5: Check the stopping criterion
Steps 3 and 4 are repeated until the termination condition is attained (maximum *NI*). The best harmony vector in HM is used to reshape the input image and obtain the quantized image.

4.3. Hybrid Color Quantization Algorithm Based on KM and HS (KMHS-CQ)

The second proposed algorithm is the KMHS-CQ. The algorithm consists of two main stages. In the first stage, the KM algorithm is applied to the compressed color space to generate the initial values for cluster centroids. These centroids are used to seed the initial harmony matrix. The *replacement parameter* $\psi \in$[0, 1] determines the friction of harmony vectors in the initial randomly-generated harmony matrix that gets replaced by the KM algorithm. When the ψ= 0 the entire HM is generated randomly. When ψ= 1 the whole HM is generated by the KM algorithm. In the second stage, HS will be applied to obtain an optimal solution for the problem. The proposed algorithm can be described by the flowchart shown in Figure 1.

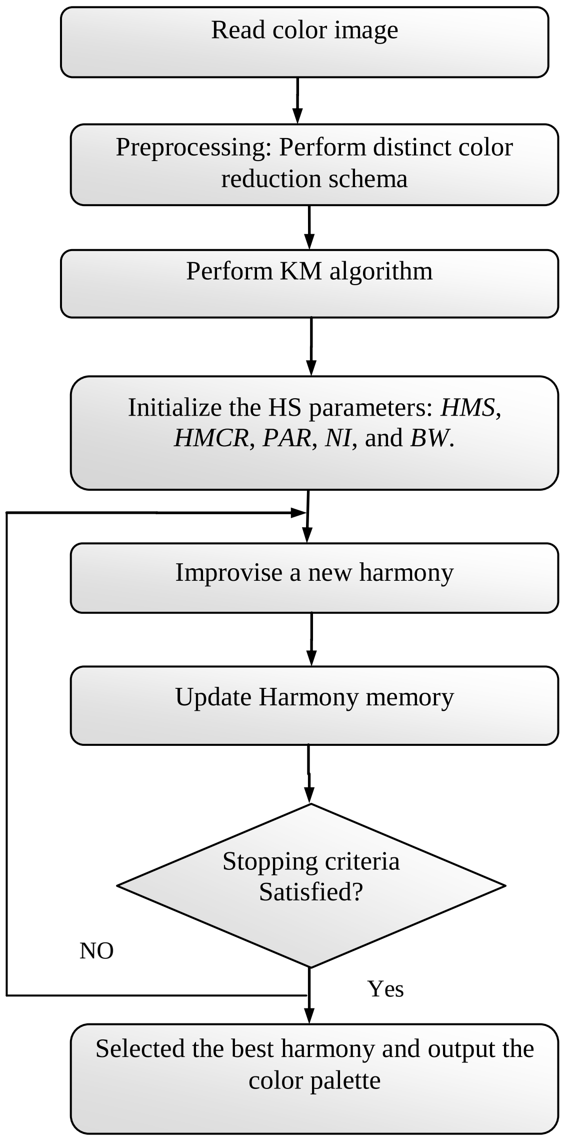
Read color image
Preprocessing: Perform distinct color reduction schema
Perform KM algorithm
Initialize the HS parameters: *HMS*, *HMCR*, *PAR*, *NI*, and *BW*.
Improvise a new harmony
Update Harmony memory
Stopping criteria Satisfied?
NO
Yes
Selected the best harmony and output the color palette

Figure.1. The proposed KMHS-CQ algorithm

This framework can achieve a good balance between computational efficiency and quantization quality.

5. EXPERIMENTAL RESULTS

To evaluate the efficiency of the proposed algorithm experimental tests are executed on a Pentium 4 3.40 GHz processor with 2 GB RAM. The HS and KMHS algorithms are implemented using MATLAB7.1 and tested on a set of four commonly used test images in the quantization literature [23]. Each test image is a true-color image of 512 × 512 pixels. These images are:

Lenna with 148,279 (56.6%) unique colors, *Peppers* with 183, 525 (70%) unique colors, Mandrill with 230,427 colors (87.9%) unique colors, and *Airplane* with 77,041 (29.4%) unique colors as shown in Figure 2.

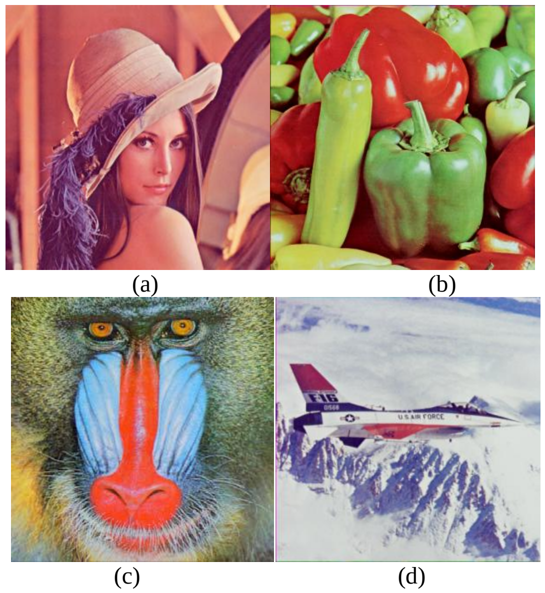

Figure.2. Original 512 x 512 test images: (a) Lenna; (b) Peppers; (c) Mandrill; (d) Airplane.

In the simulation, palette sizes of 8, 16, 32 and 64 evaluated by comparing their results with some other popular CQ algorithms reported in the literature such as: KM, Accelerated K-means (AKM) [13], mean-cut [13], Firefly and K-means hybrid (FA-KM) [20], the growing self-organizing model (GNG) [26-27], and PSO [18]. Due to the random nature of these algorithms, the results reported in this section are the average values over 5 simulations. Several simulations were performed to find the algorithmic parameters for KM, PSO, and HS algorithms. The values of *HMCR*, *PAR* were selected empirically based on the suggestions provided in [24]. The recommended range for the *HMCR* is between [0, 1] and the *PAR* between [0, 1]. The parameters of HS-CQ were set as follows: *HMS*=5, *HMCR*=0.98, *PAR*=0.99, *BW*=0.8 and *NI*=5000 iterations. The KMHS-CQ parameters were set as follows: $\psi = 0.2$ and the same parameters as of HS-CQ were used but *NI* was reduced to 50 iterations. For the KM the maximum number of iterations was set to 200. For PSO we adopted the following parameters: the swarm size *N*=25, the inertia weights w_1=0.9 and w_2=0.4, the acceleration constants c_1=c_2=2.0, the maximum velocity V_{max}=4 and the maximum number of iteration NI_{max}=200. The efficiency of the CQ algorithm is measured by the MSE and PSNR calculated as:

$$\boldsymbol{MSE(I,I')} = \frac{1}{3mn}\sum_{i}^{n}\sum_{j}^{m}\Big[\big(R(i,j) - R'^{(i,j)}\big)^2 + \big(G(i,j) - G'^{(i,j)}\big)^2 + \big(B(i,j) - B'^{(i,j)}\big)^2\Big], \qquad (8)$$

$$\boldsymbol{PSNR(I,I')} = 10\,log_{10}\frac{255^2}{MSE(I,I')}, \qquad (9)$$

where *I* is original image and *I'* is the quantized image. *R(i, j)*,*G(i, j)* and *B(i, j)*are the red, green and blue pixel values respectively. Figure 3 presents the resultant quantized images using the KMHS-CQ algorithm with 8, 16, 32, and 64 palette colors. Blocking and the stair case effects can be visually observed in compressed images with palettes of 8 and 16 colors. This is due to the fact that these palette colors are not sufficient for characterizing all the color pixels in the test images. However, good quantization quality is achieved with 32 and 64 colors palettes as it is hard to distinguish them from the original images. To understand the quantitative qualities of the compressed images, PSNR values' of KMHS-CQ for the 8, 16, 32, and 64 colors quantized images are shown in Figure 4. Generally, the quantized image quality increases with the increment of the palette size. For example, the *Lenna* image achieves PNSR values of 26.886, 29.702, 32.2, and 34.36 for the 8, 16, 32 and 64 color palettes, respectively. The performance of the proposed algorithms is compared to other CQ algorithms in the literature such as KM [13], AKM [13], MC [13], FA-KM [20], GNG [26-27] and PSO [18]. Comparative results of the average PNSR values of the different algorithms are presented in Table1. The best (highest) PNSR values are shown in ***bold***. According to the results given in Table 1,the KMHS method achieves the best image quality for almost all test cases. KMHS achieved the highest PNSR values in all test conditions except *Mandrill* (*k*=8) where it achieved a PNSR value of 22.423 in contrast to 22.425 by FA-KM. Similarly, for Airplane (*k*=16) KMHS achieved a PNSR of 31.622 in contrast to 31.866 by AKM and 32.014 by HS. KMHS provided slightly better image quality than that of the HS algorithm in all listed cases except for Airplane (*k*=16). FA-KM achieved similar results to KMHS in *Lenna* (*k*=8), *Peppers* (*k*=8), *Mandrill* (*k*=16), and *Mandrill* (*k*=32).

Experimental results of the execution time of the two proposed schemes are given in Table 2. The required execution time generally increases when a larger *k* value is used. The processing time of KMHS is much less than the stand-alone HS.

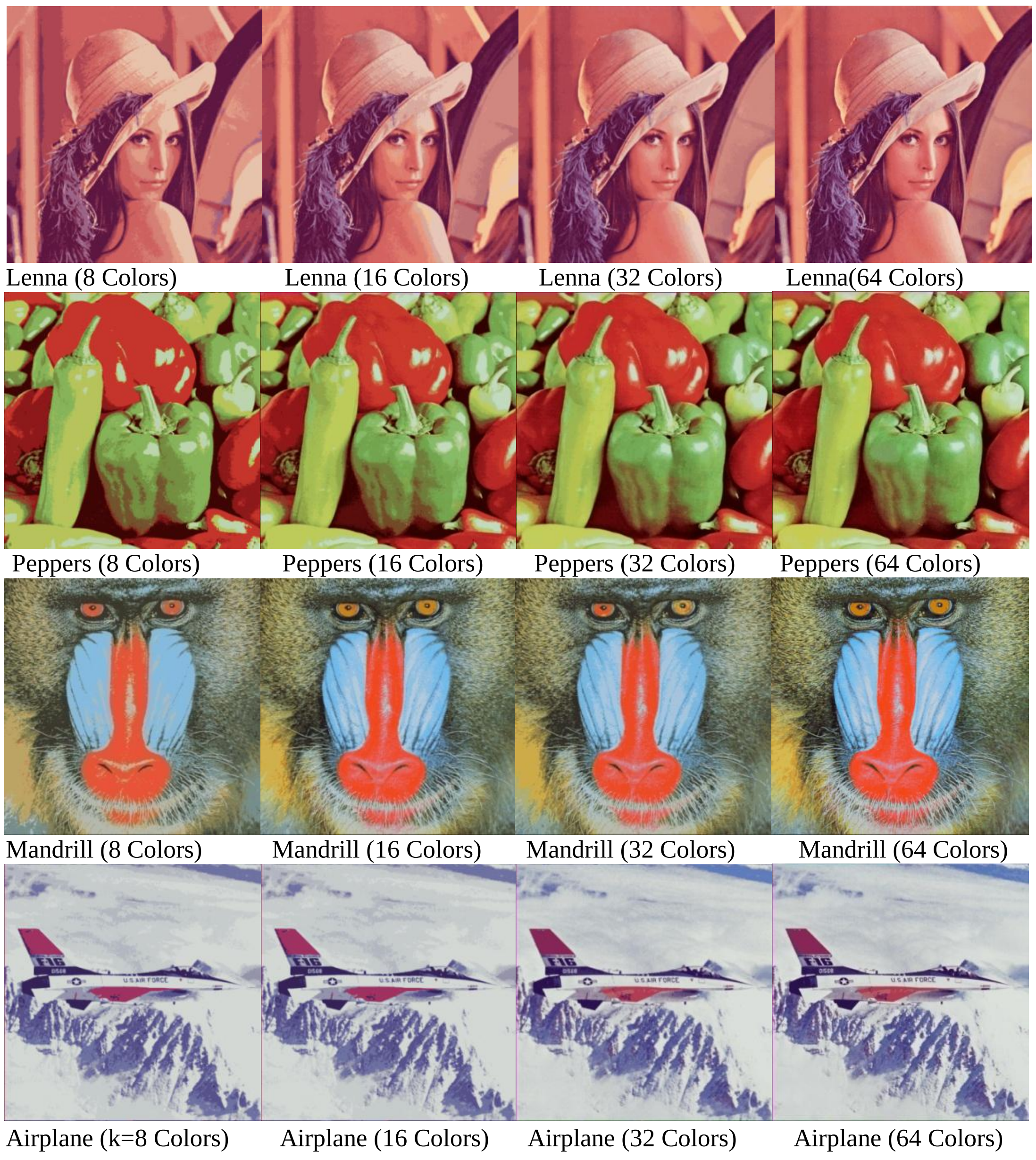

Figure.3. The quantized images obtained by KMHS-CQ for 8, 16, 32 and 64 color palettes.

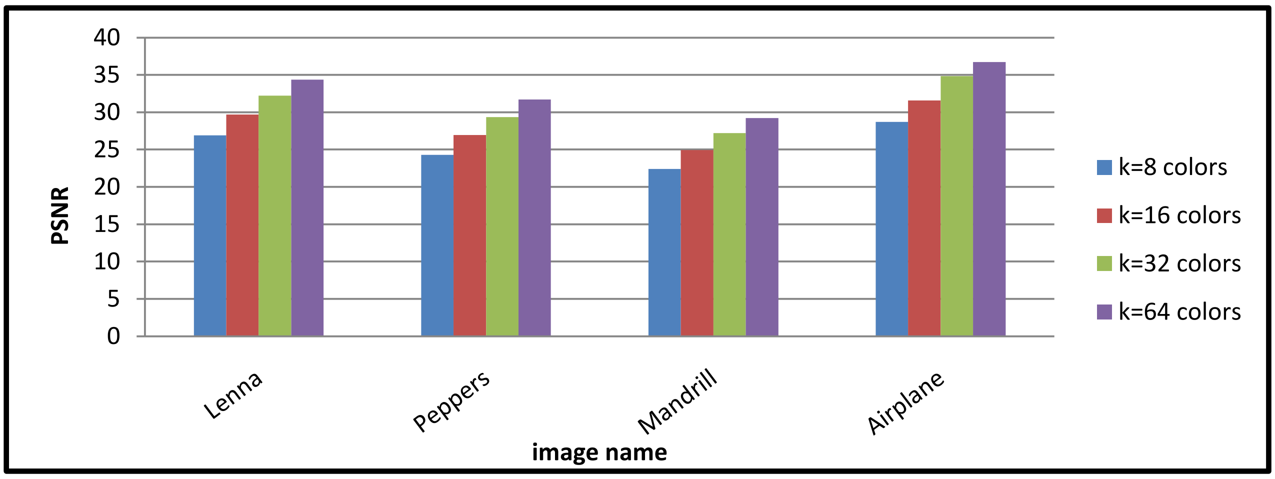

Figure.4. PSNR (in dB) obtained by KMHS-CQ for 8, 16, 32, and 64 color palettes in (a) Lenna; (b) Pepper; (c) Mandrill; (d) Airplane.

Table 1.Average PNSR (in dB)for different CQ algorithms with different palette sizes for the different test images. N/A stands for "not available".

Image	K	Algorithms							
		KM	*AKM[13]*	*MC[13]*	*FA-KM[20]*	*GNG[26-27]*	*PSO-CQ[18]*	*HS-CQ*	*KMHS-CQ*
Lenna	**8**	26.722	N/A	N/A	**26.886**	26.75	26.782	26.885	**26.886**
	16	29.553	29.452	28.910	29.700	29.54	28.740	29.641	**29.702**
	32	32.161	31.815	31.331	32.199	32.05	30.251	31.949	**32.200**
	64	34.182	33.381	33.509	N/A	34.21	31.249	33.314	**34.360**
Peppers	**8**	24.275	N/A	N/A	**24.307**	24.16	24.199	24.290	**24.307**
	16	26.669	26.661	25.971	26.853	26.60	26.631	26.835	**26.902**
	32	29.286	29.069	28.096	29.321	28.99	28.121	29.236	**29.326**
	64	31.220	31.493	30.381	N/A	31.37	29.303	30.906	**31.683**
Mandrill	**8**	22.394	N/A	N/A	**22.425**	22.31	22.411	22.422	22.423
	16	24.887	N/A	N/A	**24.923**	24.78	24.737	24.906	**24.923**
	32	27.151	N/A	N/A	**27.191**	27.07	26.486	27.120	**27.191**
	64	29.183	N/A	N/A	N/A	29.07	28.028	28.663	**29.210**
Airplane	**8**	28.694	N/A	N/A	N/A	28.57	28.500	28.642	28.695
	16	30.845	31.866	29.882	N/A	30.83	31.038	**32.014**	31.622
	32	33.652	34.519	31.883	N/A	34.62	32.844	34.508	**34.766**
	64	36.345	36.242	35.384	N/A	36.20	33.704	35.877	**36.729**

Table 2. Comparison between execution time (in Seconds) for HS-CQ and KMHS-CQ.

Image	Palette Size							
	K=8		K=16		K=32		K=64	
	HS-CQ	KMHS-CQ	HS-CQ	KMHS-CQ	HS-CQ	KMHS-CQ	HS-CQ	KMHS-CQ
Lenna	313	15	636	20	1107	29	2341	57
Peppers	369	14	690	25	1478	60	2900	77
Mandrill	469	14	1003	25	1920	66	3492	75
Airplane	181	13	352	34	629	47	1117	75

6. CONCLUSION

In this paper, new organized study by the proposed algorithms for color image quantization based on the Harmony Search (HS) clustering algorithm have been investagated. Two variants of the algorithm are examined. The first is based on a stand-alone HS algorithm and the second is a hybrid algorithm of K-means and HS. The combination of the two algorithms can overcome the drawbacks of each of the two algorithms. A preprocessing color reduction scheme is applied to reduce the number of distinct colors in the input image before performing the clustering phase. Performance of the proposed algorithms has been measured using the peak signal-to-noise ratio (PSNR). Experimental results using a set of commonly-used test images in the quantization literature have demonstrated that the proposed algorithms are efficient heuristics that outperform state-of-the-art quantization methods with respect to distortion minimization.

7. REFERENCES

[1] Z. Xiang and G. Joy. " Color image quantization by agglomerative clustering." IEEE Computer Graphics and Applications 14.3 (1994). Pages: 44-48.

[2] J. P. Braquelaire and L. Brun, " Comparison and optimization of methods of color image quantization." Image Processing, IEEE Transactions on 6.7 (1997). Pages: 1048-1052.

[3] P. Scheunders. " A comparison of clustering algorithms applied to color image quantization." Pattern Recognition Letters 18.11 (1997). Pages: 1379-1384.

[4] P. Scheunders. " A genetic c-means clustering algorithm applied to color image quantization." Pattern Recognition 30.6 (1997). Pages: 859-866

[5] P. Heckbert. " Color image quantization for frame buffer display" ACM 16. 3 (1982).

[6] S. J. Wan and P. Prusinkiewicz, and S. K. M. Wong. " Variance based color image quantization for frame buffer display." Color Research & Application 15.1 (1990). Pages: 52-58.

[7] G. Joy and Z. Xiang, " Center-cut for color-image quantization."The Visual Computer 10.1 (1993). Pages: 62-66.

[8] M. Gervautz and W. Purgathofer. " A simple method for color quantization: Octree quantization." New trends in computer graphics.Springer Berlin Heidelberg, 1988. Pages: 219-231.

[9] M. T. Orchard and C. A. Bouman, " Color quantization of images."Signal Processing, IEEE Transactions on 39.12 (1991). PAges: 2677-2690.

[10] X. Wu. " Efficient statistical computations for optimal color quantization,"Graphics gems 2 (1991). Pages: 126-133.

[11] C. Y. Yang, and J. C. Lin. " RWM-cut for color image quantization," Computers & graphics 20.4 (1996). Pages: 577-588.

[12] Z. Xiang. " Color image quantization by minimizing the maximum intercluster distance." ACM Transactions on Graphics (TOG) 16.3 (1997). Pages: 260-276.

[13] Y. C. Hu and B. H. Su. " Accelerated k-means clustering algorithm for colour image quantization." The Imaging Science Journal 56.1 (2008). Pages: 29-40.

[14] M. E. Celebi. " An Effective Color Quantization Method Based on the Competitive Learning Paradigm."IPCV. 2009. Pages: 876-880.

[15] Q. Wen and M. E. Celebi. " Hard versus fuzzy c-means clustering for color quantization."EURASIP Journal on Advances in Signal Processing 2011.1 (2011). Pages: 1-12.

[16] A. H. Dekker. " Kohonen neural networks for optimal colour quantization." Network: Computation in Neural Systems 5.3 (1994). Pages: 351-367.

[17] T. Taşdizen, L. Akarun and C. Ersoy,. " Color quantization with genetic algorithms."Signal Processing: Image Communication 12.1 (1998). Pages: 49-57.

[18] M. G. Omran, A. P. Engelbrecht and A. Salman. " A color image quantization algorithm based on particle swarm optimization." INFORMATICA-LJUBLJANA-29.3 (2005). Pages: 261.

[19] R. Kaur, A. Girdhar and S. Gupta, " Color Image Quantization based on Bacteria Foraging Optimization." International Journal of Computer Applications (0975–8887) Volume (2011).

[20] P. Jitpakdee, P. Aimmanee and B. Uyyanonvara. " A Hybrid Approach for Color Image Quantization Using K-means and Firefly Algorithms." World Academy of Science, Engineering and Technology 77 (2013). Pages: 138-145.

[21] R. Kaur and V. Gupta. " Proposed Method for Color Image Quantization: Honey Bee."International Journal of Computer Science and Communication Engineering 1.2(2012). Pages: 19-22.

[22] Q. Su, S. Guo, Z . Huang and Z. Hu. " Color Image Quantization Algorithm Based on Differential Evolution." Journal of Software 8.12 (2013). Pages: 3035-3041.

[23] A. G. Weber. " The USC-SIPI image database." Signal and Image Processing Institute of the University of Southern California. URL: http://sipi. usc. edu/services/database (1997).

[24] Z. W. Geem, J. H. Kim and G. V. Loganathan " A new heuristic optimization algorithm: harmony search." Simulation 76.2 (2001). Pages: 60-68.

[25] A. Mohabey and A. K. Ray. " Rough set theory based segmentation of color images." Fuzzy Information Processing Society, 2000.NAFIPS.19th International Conference of the North American. IEEE, 2000. Pages: 338-342.

[26] E. J. Palomo and E. Domínguez " Hierarchical Color Quantization Based on Self-organization."Journal of mathematical imaging and vision 49.1 (2014): 1-19.

[27] A. Atsalakis and N. Papamarkos. " Color reduction and estimation of the number of dominant colors by using a self-growing and self-organized neural gas."Engineering Applications of Artificial Intelligence 19.7 (2006). Pages: 769-786.