# Fast color quantization using weighted sort-means clustering

**M. Emre Celebi**

*Department of Computer Science, Louisiana State University in Shreveport, Shreveport, Louisiana 71115, USA (ecelebi@lsus.edu)*

Received April 22, 2009; revised July 24, 2009; accepted August 15, 2009; posted September 1, 2009 (Doc. ID 110351); published October 27, 2009

Color quantization is an important operation with numerous applications in graphics and image processing. Most quantization methods are essentially based on data clustering algorithms. However, despite its popularity as a general purpose clustering algorithm, $K$-means has not received much respect in the color quantization literature because of its high computational requirements and sensitivity to initialization. In this paper, a fast color quantization method based on $K$-means is presented. The method involves several modifications to the conventional (batch) $K$-means algorithm, including data reduction, sample weighting, and the use of the triangle inequality to speed up the nearest-neighbor search. Experiments on a diverse set of images demonstrate that, with the proposed modifications, $K$-means becomes very competitive with state-of-the-art color quantization methods in terms of both effectiveness and efficiency. © 2009 Optical Society of America

*OCIS codes:* 100.2000, 100.5010.

## 1. INTRODUCTION

True-color images typically contain thousands of colors, which makes their display, storage, transmission, and processing problematic. For this reason, color quantization (reduction) is commonly used as a preprocessing step for various graphics and image processing tasks. In the past, color quantization was a necessity due to the limitations of the display hardware, which could not handle the 16 million possible colors in 24-bit images. Although 24-bit display hardware has become more common, color quantization still maintains its practical value [1]. Modern applications of color quantization include (i) image compression [2], (ii) image segmentation [3], (iii) image analysis [4], (iv) image watermarking [5], and (v) content-based image retrieval [6].

The process of color quantization is mainly composed of two phases: palette design (the selection of a small set of colors that represents the original image colors) and pixel mapping (the assignment of each input pixel to one of the palette colors). The primary objective is to reduce the number of unique colors, $N'$, in an image to $K$ ($K \ll N'$) with minimal distortion. In most applications, 24-bit pixels in the original image are reduced to 8 bits or fewer. Since natural images often contain a large number of colors, faithful representation of these images with a limited palette is a difficult problem.

Color quantization methods can be broadly classified into two categories [7]: image-independent methods that determine a universal (fixed) palette without regard to any specific image [8], and image-dependent methods that determine a custom (adaptive) palette based on the color distribution of the images. Despite being very fast, image-independent methods usually give poor results since they do not take into account the image contents. Therefore, most of the studies in the literature consider only image-dependent methods, which strive to achieve a better balance between computational efficiency and visual quality of the quantization output.

Numerous image-dependent color quantization methods have been developed in the past three decades. These can be categorized into two families: preclustering methods and postclustering methods [1]. Preclustering methods are mostly based on the statistical analysis of the color distribution of the images. Divisive preclustering methods start with a single cluster that contains all $N$ image pixels. This initial cluster is recursively subdivided until $K$ clusters are obtained. Well-known divisive methods include median cut [9], octree [10], variance-based method [11], binary splitting [12], greedy orthogonal bipartitioning [13], center cut [14], and radius-weighted-mean cut [15]. More recent methods can be found in [16–18]. On the other hand, agglomerative preclustering methods [19–23] start with $N$ singleton clusters, each of which contains one image pixel. These clusters are repeatedly merged until $K$ clusters remain. In contrast to preclustering methods that compute the palette only once, postclutering methods first determine an initial palette and then improve it iteratively. Essentially, any data clustering method can be used for this purpose. Since these methods involve iterative or stochastic optimization, they can obtain higher-quality results when compared with preclustering methods, at the expense of increased computational time. Clustering algorithms adapted to color quantization include $K$-means (KM) [24–27], minmax [28], competitive learning [29–31], fuzzy $c$-means [32,33], BIRCH [34], and self-organizing maps [35–37].

In this paper, a fast color quantization method based on the KM clustering algorithm [38] is presented. The method first reduces the amount of data to be clustered by sampling only the pixels with unique colors. In order to incorporate the color distribution of the pixels into the clustering procedure, each color sample is assigned a weight proportional to its frequency. These weighted samples are then clustered by using a fast and exact variant of the KM algorithm. The set of final cluster centers is taken as the quantization palette.

The rest of the paper is organized as follows. Section 2 describes the conventional KM clustering algorithm and the proposed modifications. Section 3 describes the experimental setup and presents the comparison of the proposed method with other color quantization methods. Finally, Section 4 gives the conclusions.

## 2. COLOR QUANTIZATION USING *K*-MEANS CLUSTERING ALGORITHM

The KM algorithm is inarguably one of the most widely used methods for data clustering [39]. Given a data set $X=\{\mathbf{x}_1,\dots,\mathbf{x}_N\}\in\mathbb{R}^D$, the objective of KM is to partition $X$ into $K$ exhaustive and mutually exclusive clusters $S=\{S_1,\dots,S_k\}$, $\cup_{k=1}^{K}S_k=X$, $S_i\cap S_j\equiv\emptyset$ for $i\neq j$ by minimizing the sum of squared error (SSE):

$$\mathrm{SSE}=\sum_{k=1}^{K}\sum_{\mathbf{x}_i\in S_k}\|\mathbf{x}_i-\mathbf{c}_k\|_2^2, \tag{1}$$

where $\|\;\|_2$ denotes the Euclidean ($L_2$) norm and $\mathbf{c}_k$ is the center of cluster $S_k$ calculated as the mean of the points that belong to this cluster. This problem is known to be computationally intractable even for $K=2$ [40], but a heuristic method developed by Lloyd [41] offers a simple solution. Lloyd's algorithm starts with $K$ arbitrary centers, typically chosen uniformly at random from the data points [42]. Each point is then assigned to the nearest center, and each center is recalculated as the mean of all points assigned to it. These two steps are repeated until a predefined termination criterion is met. The pseudocode for this procedure is given in Algorithm 1 (bold symbols denote vectors). Here, $m[i]$ denotes the membership of point $\mathbf{x}_i$, i.e., index of the cluster center that is nearest to $\mathbf{x}_i$.

**Algorithm 1**. Conventional K-Means Algorithm

**input** : $X=\{\mathbf{x}_1,\dots,\mathbf{x}_N\}\in\mathbb{R}^D$ ($N\times D$ input data set)
**output**: $C=\{\mathbf{c}_1,\dots,\mathbf{c}_K\}\in\mathbb{R}^D$ ($K$ cluster centers)
Select a random subset $C$ of $X$ as the initial set of cluster centers;
**while** *termination criterion is not met* **do**
  **for** $(i=1;i\le N;i=i+1)$ **do**
    Assign $\mathbf{x}_i$ to the nearest cluster;
    $m[i]=\underset{k\in\{1,\dots,K\}}{\mathrm{argmin}}\|\mathbf{x}_i-\mathbf{c}_k\|^2$;
  **end**
  Recalculate the cluster centers;
  **for** $(k=1;k\le K;k=k+1)$ **do**
    Cluster $S_k$ contains the set of points $\mathbf{x}_i$ that are nearest to the center $\mathbf{c}_k$;
    $S_k=\{\mathbf{x}_i\,|m[i]=k\}$;
    Calculate the new center $\mathbf{c}_k$ as the mean of the points that belong to $S_k$;
    $\mathbf{c}_k=\frac{1}{|S_k|}\sum_{\mathbf{x}_i\in S_k}\mathbf{x}_i$;
  **end**
**end**

When compared with the preclustering methods, there are two problems with using KM for color quantization. First, because of its iterative nature, the algorithm might require an excessive amount of time to obtain an acceptable output quality. Second, the output is quite sensitive to the initial choice of the cluster centers. In order to address these problems, we propose several modifications to the conventional KM algorithm:

- **Data sampling.** A straightforward way to speed up KM is to reduce the amount of data, which can be achieved by sampling the original image. Although random sampling can be used for this purpose, there are two problems with this approach. First, random sampling will further destabilize the clustering procedure in the sense that the output will be less predictable. Second, the sampling rate will be an additional parameter that will have a significant impact on the output. In order to avoid these drawbacks, we propose a deterministic sampling strategy in which only the pixels with unique colors are sampled. The unique colors in an image can be determined efficiently by using a hash table that uses chaining for collision resolution and a universal hash function of the form $h_a(\mathbf{x})=(\Sigma_{i=1}^{3}a_ix_i)$ mod $m$, where $\mathbf{x}=(x_1,x_2,x_3)$ denotes a pixel with red ($x_1$), green ($x_2$), and blue ($x_3$) components, $m$ is a prime number, and the elements of sequence $a=(a_1,a_2,a_3)$ are chosen randomly from the set $\{0,1,\dots,m-1\}$.
- **Sample weighting.** An important disadvantage of the proposed sampling strategy is that it disregards the color distribution of the original image. In order to address this problem, each point is assigned a weight that is proportional to its frequency (note that the frequency information is collected during the data sampling stage). The weights are normalized by the number of pixels in the image to avoid numerical instabilities in the calculations. In addition, Algorithm 1 is modified to incorporate the weights into the clustering procedure.
- **Sort-means (SM) algorithm.** The assignment phase of KM involves many redundant distance calculations. In particular, for each point, the distances to each of the $K$ cluster centers are calculated. Consider a point $\mathbf{x}_i$, two cluster centers $\mathbf{c}_a$ and $\mathbf{c}_b$, and a distance metric $d$; using the triangle inequality, we have $d(\mathbf{c}_a,\mathbf{c}_b)\le d(\mathbf{x}_i,\mathbf{c}_a)+d(\mathbf{x}_i,\mathbf{c}_b)$. Therefore, if we know that $2d(\mathbf{x}_i,\mathbf{c}_a)\le d(\mathbf{c}_a,\mathbf{c}_b)$, we can conclude that $d(\mathbf{x}_i,\mathbf{c}_a)\le d(\mathbf{x}_i,\mathbf{c}_b)$ without having to calculate $d(\mathbf{x}_i,\mathbf{c}_b)$. The compare-means algorithm [43] precalculates the pairwise distances between cluster centers at the beginning of each iteration. When searching for the nearest cluster center for each point, the algorithm often avoids a large number of distance calculations with the help of the triangle inequality test. The SM algorithm [43] further reduces the number of distance calculations by sorting the distance values associated with each cluster center in ascending order. At each iteration, point $\mathbf{x}_i$ is compared against the cluster centers in increasing order of distance from the center $\mathbf{c}_k$ that $\mathbf{x}_i$ was assigned to in the previous iteration. If a center that is far enough from $\mathbf{c}_k$ is reached, all of the remaining centers can be skipped, and the procedure continues with the next point. In this way, SM avoids the overhead of going through all the centers. It should be noted that more elaborate approaches to accelerate KM have been proposed in the literature. These include algorithms based on *Kd*-trees [44], core sets [45], and more sophisticated uses of the triangle inequality [46]. Some of these algorithms [45,46] are not suitable for low-dimensional data sets

such as color image data, since they incur significant overhead to create and update auxiliary data structures [46]. Others [44] provide computational gains comparable with SM at the expense of significant conceptual and implementation complexity. In contrast, SM is conceptually simple, easy to implement, and incurs very small overhead, which makes it an ideal candidate for color clustering.

We refer to the KM algorithm with the abovementioned modifications as the weighted SM (WSM) algorithm. The pseudocode for WSM is given in Algorithm 2.

**Algorithm 2.** Weighted Sort-Means Algorithm

```
input  : X = {x_1, ..., x_N'} ∈ R^D (N' × D input data set)
         W = {w_1, ..., w_N'} ∈ [0,1] (N' point weights)
output: C = {c_1, ..., c_K} ∈ R^D (K cluster centers)
Select a random subset C of X as the initial set of cluster centers;
while termination criterion is not met do
    Calculate the pairwise distances between the cluster centers;
    for (i = 1; i ≤ K; i = i + 1) do
        for (j = i + 1; j ≤ K; j = j + 1) do
            d[i][j] = d[j][i] = ||c_i − c_j||^2;
        end
    end
    Construct a K × K matrix M in which row i is a permutation of
    1,... K that represents the clusters in increasing order of
    distance of their centers from c_i;
    for (i = 1; i ≤ N'; i = i + 1) do
        Let S_p be the cluster that x_i was assigned to in the previous
        iteration;
        p = m[i];
        min_dist = prev_dist = ||x_i − c_p||^2;
        Update the nearest center if necessary;
        for (j = 2; j ≤ K; j = j + 1) do
            t = M[p][j];
            if d[p][t] ≥ 4 prev_dist then
                There can be no other closer center. Stop checking;
                break;
            end
            dist = ||x_i − c_t||^2;
            if dist ≤ min_dist then
                c_t is closer to x_i than c_p;
                min_dist = dist;
                m[i] = t;
            end
        end
    end
    Recalculate the cluster centers;
    for (k = 1; k ≤ K; k = k + 1) do
        Calculate the new center c_k as the weighted mean
        of points that are nearest to it;
        c_k = ( Σ_{m[i]=k} w_i x_i ) / Σ_{m[i]=k} w_i;
    end
end
```

## 3. EXPERIMENTAL RESULTS AND DISCUSSION

### A. Image Set and Performance Criteria

The proposed method was tested on some of the most commonly used test images in the quantization literature (see Fig. 1). The natural images in the set included Airplane (512 × 512, 77,041 (29%) unique colors), Baboon (512 × 512, 153,171 (58%) unique colors), Boats (787 × 576, 140,971 (31%) unique colors), Lenna (512 × 480, 56,164 (23%) unique colors), Parrots (1536 × 1024, 200,611 (13%) unique colors), and Peppers (512 × 512, 111,344 (42%) unique colors). The synthetic images included Fish (300 × 200, 28,170 (47%) unique colors) and Poolballs (510 × 383, 13,604 (7%) unique colors).

The effectiveness of a quantization method was quantified by the mean squared error (MSE) measure:

$$\mathrm{MSE}(\mathbf{X},\hat{\mathbf{X}}) = \frac{1}{HW}\sum_{h=1}^{H}\sum_{w=1}^{W}\|\mathbf{x}(h,w) - \hat{\mathbf{x}}(h,w)\|_2^2, \tag{2}$$

where $\mathbf{X}$ and $\hat{\mathbf{X}}$ denote, respectively, the $H \times W$ original and quantized images in the RGB color space. MSE represents the average distortion with respect to the $L_2^2$ norm (1) and is the most commonly used evaluation measure in the quantization literature [1,7]. Note that the peak signal-to-noise ratio (PSNR) measure can be easily calculated from the MSE value:

$$\mathrm{PSNR} = 20\log_{10}\left(\frac{255}{\sqrt{\mathrm{MSE}}}\right). \tag{3}$$

The efficiency of a quantization method was measured by CPU time in milliseconds. Note that only the palette generation phase was considered, since this is the most time-consuming part of the majority of quantization methods. All of the programs were implemented in the C language, compiled with the gcc v4.2.4 compiler, and executed on an Intel Core 2 Quad Q6700 2.66 GHz machine. The time figures were averaged over 100 runs.

### B. Comparison of WSM against Other Quantization Methods

The WSM algorithm was compared with some of the well-known quantization methods in the literature:

- **Median-cut (MC)** [9]. This method starts by building a 32 × 32 × 32 color histogram that contains the original pixel values reduced to 5 bits per channel by uniform quantization. This histogram volume is then recursively split into smaller boxes until $K$ boxes are obtained. At each step, the box that contains the largest number of pixels is split along the longest axis at the median point, so that the resulting subboxes each contain approximately the same number of pixels. The centroids of the final $K$ boxes are taken as the color palette.
- **Variance-based method (WAN)** [11]. This method is similar to MC, with the exception that at each step the box with the largest weighted variance (squared error) is split along the major (principal) axis at the point that minimizes the marginal squared error.
- **Greedy orthogonal bipartitioning (WU)** [13]. This method is similar to WAN, with the exception that at each step the box with the largest weighted variance is split along the axis that minimizes the sum of the variances on both sides.
- **Neu-quant (NEU)** [35]. This method utilizes a one-dimensional self-organizing map (Kohonen neural network) with 256 neurons. A random subset of $N/f$ pixels is used in the training phase, and the final weights of the neurons are taken as the color palette. In the experiments, the highest-quality configuration, i.e., $f=1$, was used.
- **Modified minmax (MMM)** [28]. This method choses the first center $\mathbf{c}_1$ arbitrarily from the data set, and the $i$th center $\mathbf{c}_i$ ($i=2,\ldots,K$) is chosen to be the point that

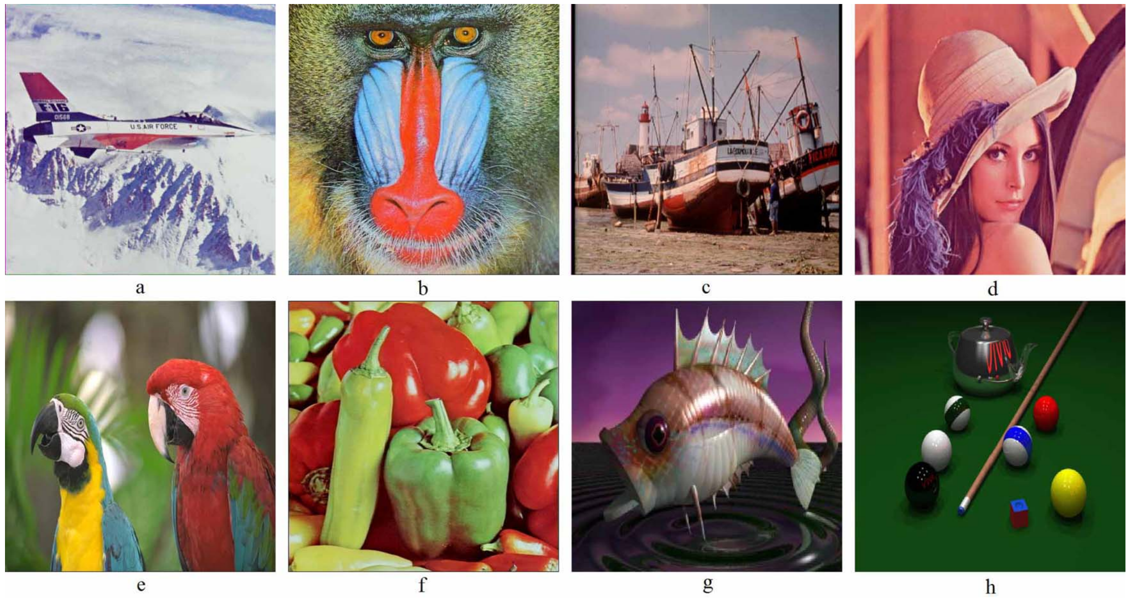

Fig. 1. (Color online) Test images: a, Airplane; b, Baboon; c, Boats; d, Lenna; e, Parrots; f, Peppers; g, Fish; h, Poolballs.

has the largest minimum weighted $L_2^2$ distance (the weights for the red, green, and blue channels are taken as 0.5, 1.0, and 0.25, respectively) to the previously selected centers, i.e., $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_{i-1}$. Each of these initial centers is then recalculated as the mean of the points assigned to it.

- **Split and Merge (SAM)** [23]. This two-phase method first divides the color space uniformly into $B$ partitions. This initial set of $B$ clusters is represented as an adjacency graph. In the second phase, $(B-K)$ merge operations are performed to obtain the final $K$ clusters. At each step of the second phase, the pair of clusters with the minimum joint quantization error are merged. In the experiments, the initial number of clusters was set to $B = 20K$.
- **Fuzzy *c*-means (FCM)** [47]. FCM is a generalization of KM in which points can belong to more than one cluster. The algorithm involves the minimization of the functional $J_q(U,V) = \sum_{i=1}^{N} \sum_{k=1}^{K} u_{ik}^q \|\mathbf{x}_i - \mathbf{v}_k\|_2^2$ with respect to $U$ (a fuzzy $K$-partition of the data set) and $V$ (a set of prototypes—cluster centers). The parameter $q$ controls the fuzziness of the resulting clusters. At each iteration, the membership matrix $U$ is updated by $u_{ik} = (\sum_{j=1}^{K} (\|\mathbf{x}_i - \mathbf{v}_k\|_2 / \|\mathbf{x}_i - \mathbf{v}_j\|_2)^{2/(q-1)})^{-1}$, which is followed by the update of the prototype matrix $V$ by $\mathbf{v}_k = (\sum_{i=1}^{N} u_{ik}^q \mathbf{x}_i) / (\sum_{i=1}^{N} u_{ik}^q)$. A näive implementation of the FCM algorithm has a complexity that is quadratic in $K$. In the experiments, a linear complexity formulation described in [48] was used, and the fuzziness parameter was set to $q = 2$, as is commonly seen in the fuzzy clustering literature [39].
- **Fuzzy *c*-means with partition index maximization (PIM)** [32]. This method is an extension of FCM in which the functional to be minimized incorporates a cluster validity measure called the partition index. This index measures how well a point $\mathbf{x}_i$ has been classified and is defined as $P_i = \sum_{k=1}^{K} u_{ik}^q$. The FCM functional can be modified to incorporate $P_i$ as follows: $J_q^{\alpha}(U,V) = \sum_{i=1}^{N} \sum_{k=1}^{K} u_{ik}^q \|\mathbf{x}_i - \mathbf{v}_k\|_2^2 - \alpha \sum_{i=1}^{N} P_i$. The parameter $\alpha$ controls the weight of the second term. The procedure that minimizes $J_q^{\alpha}(U,V)$ is identical to the one used in FCM except for the membership matrix update equation: $u_{ik} = (\sum_{j=1}^{K} [(\|\mathbf{x}_i - \mathbf{v}_k\|_2 - \alpha) / (\|\mathbf{x}_i - \mathbf{v}_j\|_2 - \alpha)]^{2/(q-1)})^{-1}$. An adaptive method to determine the value of $\alpha$ is to set it to a fraction $0 \leq \delta < 0.5$ of the distance between the nearest two centers, i.e., $\alpha = \delta \min_{i \neq j} \|\mathbf{v}_i - \mathbf{v}_j\|_2^2$. Following [32], the fraction value was set to $\delta = 0.4$.
- **Finite-state KM (FKM)** [25]. This method is a fast approximation for KM. The first iteration is the same as that of KM. In each of the subsequent iterations, the nearest center for a point $\mathbf{x}_i$ is determined from among the $K'$ ($K' \ll K$) nearest neighbors of the center that the point was assigned to in the previous iteration. When compared with KM, this technique leads to considerable computational savings, since the nearest center search is performed in a significantly smaller set of $K'$ centers rather than the entire set of $K$ centers. Following [25], the number of nearest neighbors was set to $K' = 8$.
- **Stable-flags KM (SKM)** [26]. This method is another fast approximation for KM. The first $I'$ iterations are the same as those of KM. In the subsequent iterations, the clustering procedure is accelerated by using the concepts of center stability and point activity. More specifically, if a cluster center $\mathbf{c}_k$ does not move by more than $\theta$ units (as measured by the $L_2^2$ distance) in two successive iterations, this center is classified as stable. Furthermore, points that were previously assigned to the stable centers are classified as inactive. At each iteration, only unstable centers and active points participate in the clustering procedure. Following [26], the algorithm parameters were set to $I' = 10$ and $\theta = 1.0$.

For each KM-based quantization method (except for SKM), two variants were implemented. In the first one, the number of iterations was limited to 10, which makes this variant suitable for time-critical applications. These *fixed-iteration* variants are denoted by the plain acronyms KM, FKM, and WSM. In the second variant, to obtain higher-quality results, the method was executed until it

**Table 1. MSE Comparison of the Quantization Methods**

| Method | K = 32 | | K = 64 | | K = 128 | | K = 256 | | K = 32 | | K = 64 | | K = 128 | | K = 256 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ |
| | Airplane | | | | | | | | Baboon | | | | | | | |
| MC | 124 | - | 81 | - | 54 | - | 41 | - | 546 | - | 371 | - | 248 | - | 166 | - |
| WAN | 117 | - | 69 | - | 50 | - | 39 | - | 509 | - | 326 | - | 216 | - | 142 | - |
| WU | 75 | - | 47 | - | 30 | - | 21 | - | 422 | - | 248 | - | 155 | - | 99 | - |
| NEU | 101 | - | 47 | - | 24 | - | 15 | - | 363 | - | 216 | - | 128 | - | 84 | - |
| MMM | 134 | - | 82 | - | 44 | - | 28 | - | 489 | - | 270 | - | 189 | - | 120 | - |
| SAM | 120 | - | 65 | - | 43 | - | 31 | - | 396 | - | 245 | - | 153 | - | 99 | - |
| FCM | 74 | 9 | 44 | 4 | 29 | 2 | 21 | 1 | 415 | 15 | 265 | 10 | 174 | 6 | 119 | 4 |
| PIM | 73 | 9 | 45 | 4 | 29 | 2 | 21 | 1 | 413 | 18 | 261 | 13 | 172 | 7 | 117 | 4 |
| KM | 112 | 25 | 65 | 12 | 36 | 4 | 22 | 2 | 345 | 9 | 206 | 5 | 129 | 2 | 83 | 1 |
| KM-C | 59 | 2 | 35 | 1 | 25 | 0 | 19 | 0 | 329 | 3 | 196 | 1 | **123** | 1 | 79 | 0 |
| FKM | 113 | 19 | 64 | 9 | 36 | 4 | 22 | 1 | 346 | 9 | 206 | 4 | 129 | 2 | 83 | 1 |
| FKM-C | 59 | 2 | 35 | 1 | 26 | 1 | 19 | 1 | 328 | 3 | 196 | 1 | **123** | 1 | 79 | 0 |
| SKM | 112 | 20 | 63 | 9 | 36 | 4 | 22 | 1 | 343 | 10 | 207 | 6 | 129 | 2 | 83 | 1 |
| *WSM* | 64 | 4 | 36 | 1 | 23 | 1 | 15 | 0 | 345 | 8 | 204 | 3 | 127 | 1 | 81 | 1 |
| *WSM-C* | **57** | 1 | **34** | 0 | **22** | 0 | **14** | 0 | **327** | 3 | **195** | 1 | **123** | 1 | **78** | 0 |
| | Boats | | | | | | | | Lenna | | | | | | | |
| MC | 200 | - | 126 | - | 78 | - | 57 | - | 165 | - | 94 | - | 71 | - | 47 | - |
| WAN | 198 | - | 117 | - | 71 | - | 45 | - | 159 | - | 93 | - | 61 | - | 43 | - |
| WU | 154 | - | 87 | - | 50 | - | 32 | - | 130 | - | 76 | - | 46 | - | 29 | - |
| NEU | 147 | - | 79 | - | 41 | - | 26 | - | 119 | - | 68 | - | 36 | - | 23 | - |
| MMM | 203 | - | 114 | - | 69 | - | 41 | - | 139 | - | 86 | - | 50 | - | 34 | - |
| SAM | 161 | - | 95 | - | 59 | - | 42 | - | 135 | - | 88 | - | 56 | - | 40 | - |
| FCM | 160 | 13 | 99 | 8 | 64 | 5 | 42 | 3 | 132 | 10 | 83 | 7 | 53 | 4 | 38 | 2 |
| PIM | 161 | 14 | 99 | 11 | 63 | 5 | 43 | 3 | 136 | 12 | 81 | 6 | 53 | 4 | 38 | 2 |
| KM | 135 | 11 | 78 | 5 | 47 | 3 | 30 | 1 | 106 | 5 | 61 | 2 | 38 | 1 | 24 | 0 |
| KM-C | **115** | 1 | 64 | 1 | 39 | 0 | 25 | 0 | **97** | 1 | 57 | 1 | 35 | 0 | **22** | 0 |
| FKM | 134 | 10 | 77 | 5 | 47 | 3 | 29 | 1 | 107 | 8 | 61 | 2 | 38 | 1 | 24 | 0 |
| FKM-C | 116 | 1 | 65 | 1 | 39 | 0 | 25 | 0 | **97** | 1 | 57 | 1 | 35 | 0 | **22** | 0 |
| SKM | 137 | 13 | 77 | 4 | 47 | 2 | 30 | 1 | 107 | 6 | 62 | 2 | 38 | 1 | 24 | 1 |
| *WSM* | 125 | 7 | 68 | 2 | 40 | 1 | 24 | 0 | 103 | 5 | 60 | 2 | 36 | 1 | 23 | 0 |
| *WSM-C* | **115** | 1 | **63** | 0 | **37** | 0 | **23** | 0 | **97** | 2 | **56** | 1 | **34** | 0 | **22** | 0 |
| | Parrots | | | | | | | | Peppers | | | | | | | |
| MC | 401 | - | 258 | - | 144 | - | 99 | - | 333 | - | 213 | - | 147 | - | 98 | - |
| WAN | 365 | - | 225 | - | 146 | - | 90 | - | 333 | - | 215 | - | 142 | - | 93 | - |
| WU | 291 | - | 171 | - | 96 | - | 59 | - | 264 | - | 160 | - | 101 | - | 63 | - |
| NEU | 306 | - | 153 | - | 84 | - | 47 | - | 249 | - | 151 | - | 83 | - | 55 | - |
| MMM | 332 | - | 200 | - | 117 | - | 73 | - | 292 | - | 182 | - | 113 | - | 76 | - |
| SAM | 276 | - | 160 | - | 94 | - | 60 | - | 268 | - | 161 | - | 100 | - | 64 | - |
| FCM | 297 | 19 | 178 | 14 | 107 | 5 | 69 | 2 | 272 | 15 | 179 | 7 | 120 | 4 | 84 | 3 |
| PIM | 295 | 21 | 175 | 12 | 107 | 5 | 69 | 2 | 266 | 14 | 176 | 7 | 119 | 5 | 84 | 3 |
| KM | 262 | 20 | 149 | 9 | 85 | 4 | 51 | 2 | 232 | 7 | 141 | 4 | 87 | 2 | 54 | 1 |
| KM-C | 237 | 7 | 131 | 3 | 76 | 1 | 46 | 1 | 220 | 2 | 132 | 1 | **80** | 0 | 51 | 0 |
| FKM | 264 | 21 | 150 | 10 | 87 | 4 | 51 | 2 | 231 | 6 | 142 | 4 | 86 | 2 | 55 | 1 |
| FKM-C | 237 | 7 | 132 | 3 | 77 | 2 | 47 | 1 | 220 | 2 | 132 | 2 | 81 | 1 | 51 | 0 |
| SKM | 259 | 16 | 152 | 11 | 86 | 4 | 51 | 2 | 233 | 7 | 142 | 4 | 87 | 2 | 55 | 1 |
| *WSM* | 249 | 13 | 136 | 5 | 79 | 2 | 46 | 1 | 232 | 7 | 139 | 3 | 85 | 1 | 53 | 1 |
| *WSM-C* | **232** | 6 | **128** | 2 | **74** | 1 | **43** | 0 | **219** | 2 | **131** | 1 | **80** | 1 | **50** | 0 |
| | Fish | | | | | | | | Poolballs | | | | | | | |
| MC | 276 | - | 169 | - | 107 | - | 68 | - | 136 | - | 64 | - | 38 | - | 27 | - |
| WAN | 311 | - | 208 | - | 124 | - | 77 | - | 112 | - | 59 | - | 45 | - | 38 | - |
| WU | 187 | - | 111 | - | 69 | - | 44 | - | 68 | - | 31 | - | 17 | - | 11 | - |
| NEU | 173 | - | 107 | - | 57 | - | 42 | - | 104 | - | 44 | - | 18 | - | 9 | - |
| MMM | 235 | - | 136 | - | 81 | - | 53 | - | 166 | - | 91 | - | 42 | - | 20 | - |
| SAM | 198 | - | 120 | - | 74 | - | 49 | - | 91 | - | 54 | - | 37 | - | 20 | - |
| FCM | 169 | 11 | 110 | 5 | 79 | 3 | 60 | 3 | 153 | 75 | 61 | 30 | 25 | 5 | 14 | 2 |
| PIM | 168 | 9 | 111 | 4 | 79 | 3 | 60 | 3 | 149 | 71 | 57 | 26 | 25 | 7 | 14 | 2 |
| KM | 174 | 24 | 105 | 9 | 64 | 4 | 40 | 2 | 226 | 75 | 129 | 31 | 75 | 17 | 39 | 8 |
| KM-C | 145 | 3 | 90 | 2 | 58 | 2 | 37 | 1 | 94 | 8 | 51 | 5 | 44 | 6 | 29 | 5 |
| FKM | 173 | 17 | 105 | 10 | 65 | 4 | 40 | 2 | 229 | 73 | 130 | 44 | 78 | 15 | 37 | 6 |
| FKM-C | 144 | 3 | 90 | 2 | 59 | 2 | 38 | 1 | 95 | 9 | 55 | 10 | 45 | 8 | 27 | 5 |
| SKM | 177 | 19 | 105 | 9 | 65 | 4 | 40 | 2 | 167 | 35 | 120 | 15 | 71 | 13 | 37 | 7 |
| *WSM* | 148 | 3 | 91 | 3 | 55 | 1 | 33 | 0 | 69 | 10 | 31 | 6 | 14 | 2 | **7** | 0 |
| *WSM-C* | **142** | 4 | **85** | 1 | **52** | 1 | **32** | 0 | **62** | 6 | **27** | 3 | **13** | 1 | **7** | 0 |

**Table 2. CPU Time Comparison of the Quantization Methods**

| Method | K = 32 | K = 64 | K = 128 | K = 256 | K = 32 | K = 64 | K = 128 | K = 256 |
|---|---|---|---|---|---|---|---|---|
| | Airplane | | | | Baboon | | | |
| MC | 10 | 10 | **11** | **12** | 10 | **10** | **11** | **13** |
| WAN | 13 | 14 | 15 | 18 | 14 | 15 | 16 | 20 |
| WU | 16 | 16 | 16 | 16 | 16 | 15 | 16 | 17 |
| NEU | 70 | 142 | 265 | 514 | 67 | 134 | 254 | 485 |
| MMM | 123 | 206 | 367 | 696 | 126 | 207 | 375 | 702 |
| SAM | **7** | **8** | 13 | 25 | **9** | 20 | 56 | 112 |
| FCM | 2739 | 5285 | 10612 | 21079 | 2737 | 5285 | 10612 | 21081 |
| PIM | 2410 | 5038 | 10402 | 20913 | 2488 | 5091 | 10407 | 20846 |
| KM | 584 | 1005 | 1791 | 3314 | 592 | 1012 | 1800 | 3317 |
| KM-C | 17688 | 43850 | 74814 | 71908 | 3136 | 7070 | 13164 | 25657 |
| FKM | 189 | 222 | 299 | 505 | 189 | 223 | 299 | 508 |
| FKM-C | 4111 | 6144 | 6057 | 5376 | 746 | 934 | 1171 | 1959 |
| SKM | 530 | 903 | 1593 | 2952 | 547 | 927 | 1610 | 2961 |
| *WSM* | 68 | 92 | 145 | 301 | 147 | 188 | 270 | 477 |
| *WSM-C* | 257 | 359 | 522 | 1180 | 401 | 565 | 814 | 1580 |
| | Boats | | | | Lenna | | | |
| MC | 19 | **18** | **19** | **21** | 9 | 8 | 10 | **10** |
| WAN | 24 | 24 | 26 | 29 | 12 | 15 | 15 | 17 |
| WU | 28 | 26 | 28 | 28 | 15 | 15 | 14 | 15 |
| NEU | 122 | 232 | 453 | 853 | 61 | 123 | 244 | 465 |
| MMM | 219 | 367 | 656 | 1237 | 116 | 193 | 346 | 654 |
| SAM | **17** | 19 | 21 | 32 | **8** | **7** | **9** | 13 |
| FCM | 4695 | 9141 | 18350 | 36471 | 2545 | 4954 | 9953 | 19770 |
| PIM | 4075 | 8555 | 17784 | 36071 | 2348 | 4820 | 9832 | 19681 |
| KM | 986 | 1727 | 3087 | 5729 | 536 | 939 | 1673 | 3101 |
| KM-C | 9853 | 22622 | 53858 | 111047 | 3457 | 6698 | 11927 | 23762 |
| FKM | 326 | 385 | 509 | 804 | 170 | 205 | 281 | 478 |
| FKM-C | 2393 | 3158 | 4007 | 6056 | 788 | 878 | 1167 | 1886 |
| SKM | 908 | 1551 | 2756 | 5105 | 485 | 837 | 1493 | 2778 |
| *WSM* | 136 | 174 | 255 | 464 | 52 | 68 | 110 | 244 |
| *WSM-C* | 486 | 614 | 853 | 1647 | 149 | 212 | 329 | 883 |
| | Parrots | | | | Peppers | | | |
| MC | **57** | **58** | **59** | **61** | 10 | **10** | **11** | **12** |
| WAN | 81 | 82 | 83 | 86 | 13 | 14 | 16 | 18 |
| WU | 86 | 87 | 86 | 87 | 16 | 17 | 17 | 17 |
| NEU | 476 | 849 | 1571 | 2914 | 70 | 135 | 262 | 493 |
| MMM | 758 | 1265 | 2282 | 4286 | 125 | 206 | 371 | 700 |
| SAM | 74 | 77 | 103 | 150 | **8** | 11 | 29 | 53 |
| FCM | 16096 | 31734 | 63871 | 126554 | 2739 | 5288 | 10624 | 21107 |
| PIM | 14620 | 30159 | 61891 | 124794 | 2499 | 5107 | 10425 | 20883 |
| KM | 3309 | 5918 | 10657 | 19828 | 564 | 996 | 1785 | 3309 |
| KM-C | 23949 | 61168 | 119907 | 242439 | 3387 | 7761 | 14839 | 31893 |
| FKM | 1100 | 1302 | 1698 | 2519 | 181 | 219 | 295 | 500 |
| FKM-C | 5464 | 8557 | 9529 | 10482 | 869 | 1017 | 1262 | 2233 |
| SKM | 3072 | 5429 | 9506 | 17599 | 523 | 905 | 1605 | 2971 |
| *WSM* | 250 | 298 | 399 | 639 | 107 | 138 | 201 | 373 |
| *WSM-C* | 634 | 820 | 1261 | 2149 | 327 | 466 | 648 | 1387 |
| | Fish | | | | Poolballs | | | |
| MC | 6 | **5** | **7** | **6** | **9** | **9** | **9** | **11** |
| WAN | 5 | 6 | 8 | 12 | 10 | 10 | 12 | 14 |
| WU | 8 | 9 | 8 | 9 | 12 | 13 | 12 | 13 |
| NEU | 12 | 27 | 58 | 110 | 51 | 103 | 192 | 353 |
| MMM | 23 | 34 | 59 | 112 | 87 | 145 | 263 | 498 |
| SAM | **4** | 6 | 9 | 17 | **9** | 10 | 16 | 23 |
| FCM | 610 | 1209 | 2428 | 4832 | 1999 | 3940 | 7913 | 15719 |
| PIM | 560 | 1171 | 2401 | 4806 | 1586 | 3406 | 6817 | 13257 |
| KM | 128 | 229 | 404 | 757 | 396 | 703 | 1281 | 2400 |
| KM-C | 1147 | 2777 | 4395 | 5233 | 3339 | 13294 | 14912 | 22637 |
| FKM | 39 | 49 | 78 | 187 | 133 | 158 | 213 | 369 |
| FKM-C | 267 | 346 | 420 | 893 | 913 | 1565 | 1285 | 2036 |
| SKM | 121 | 207 | 361 | 672 | 380 | 653 | 1173 | 2174 |
| *WSM* | 25 | 32 | 57 | 173 | **9** | 15 | 34 | 136 |
| *WSM-C* | 85 | 109 | 182 | 572 | 24 | 34 | 94 | 356 |

**Table 3. Performance Rank Comparison of the Quantization Methods**

| Method | MSE Rank | Time Rank | Mean Rank |
|---|---|---|---|
| MC | 13.97 | **1.38** | 7.67 |
| WAN | 13.66 | 2.84 | 8.25 |
| WU | 8.47 | 3.31 | 5.89 |
| NEU | 6.31 | 6.00 | 6.16 |
| MMM | 12.31 | 7.63 | 9.97 |
| SAM | 10.09 | 2.53 | 6.31 |
| FCM | 10.31 | 13.94 | 12.13 |
| PIM | 9.81 | 12.94 | 11.38 |
| KM | 7.56 | 11.34 | 9.45 |
| KM-C | 3.03 | 15.00 | 9.02 |
| FKM | 7.91 | 7.75 | 7.83 |
| FKM-C | 3.88 | 11.53 | 7.70 |
| SKM | 8.06 | 10.25 | 9.16 |
| *WSM* | 3.56 | 5.28 | **4.42** |
| *WSM-C* | **1.06** | 8.25 | 4.66 |

**Table 4. Stability Rank Comparison of the Quantization Methods**

| Method | MSE Rank |
|---|---|
| FCM | 9.36 |
| PIM | 9.56 |
| KM | 8.31 |
| KM-C | 2.84 |
| FKM | 8.10 |
| FKM-C | 3.41 |
| SKM | 7.11 |
| *WSM* | 3.92 |
| *WSM-C* | **2.02** |

converged. Convergence was determined by the following commonly used criterion [38]: $(\mathrm{SSE}_{i-1}-\mathrm{SSE}_i)/\mathrm{SSE}_i \leq \epsilon$, where $\mathrm{SSE}_i$ denotes the SSE [Eq. (1)] value at the end of the $i$th iteration. Following [25,26], the convergence threshold was set to $\epsilon = 0.0001$. The *convergent* variants of KM, FKM, and WSM are denoted KM-C, FKM-C, and WSM-C, respectively. Note that since SKM involves at least $I' = 10$ iterations, only the convergent variant was implemented for this method. As for the fuzzy quantization methods, i.e., FCM and PIM, because of their excessive computational requirements, the number of iterations for these methods was limited to 10.

Tables 1 and 2 compare the performance of the methods at quantization levels $K = \{32, 64, 128, 256\}$ on the test images. Note that, for computational simplicity, random initialization was used in the implementations of FCM, PIM, KM, KM-C, FKM, FKM-C, SKM, WSM, and WSM-C. Therefore, in Table 1, the quantization errors for these methods are specified in the form of mean (μ) and standard deviation ($\sigma$) over 100 runs. The best (lowest) error values are shown in bold. In addition, with respect to each performance criterion, the methods are ranked based on their mean values over the test images. Table 3 gives the mean ranks of the methods. The last column gives the overall mean ranks with the assumption that each criterion has equal importance. Note that the best possible rank is 1. The following observations are in order:

- In general, the postclustering methods are more effective but less efficient than the preclustering methods.
- With respect to distortion minimization, WSM-C outperforms the other methods by a large margin. This method obtains an MSE rank of 1.06, which means that it almost always obtains the lowest distortion.
- WSM obtains a significantly better MSE rank than its fixed-iteration rivals.
- Overall, WSM and WSM-C are the best methods.

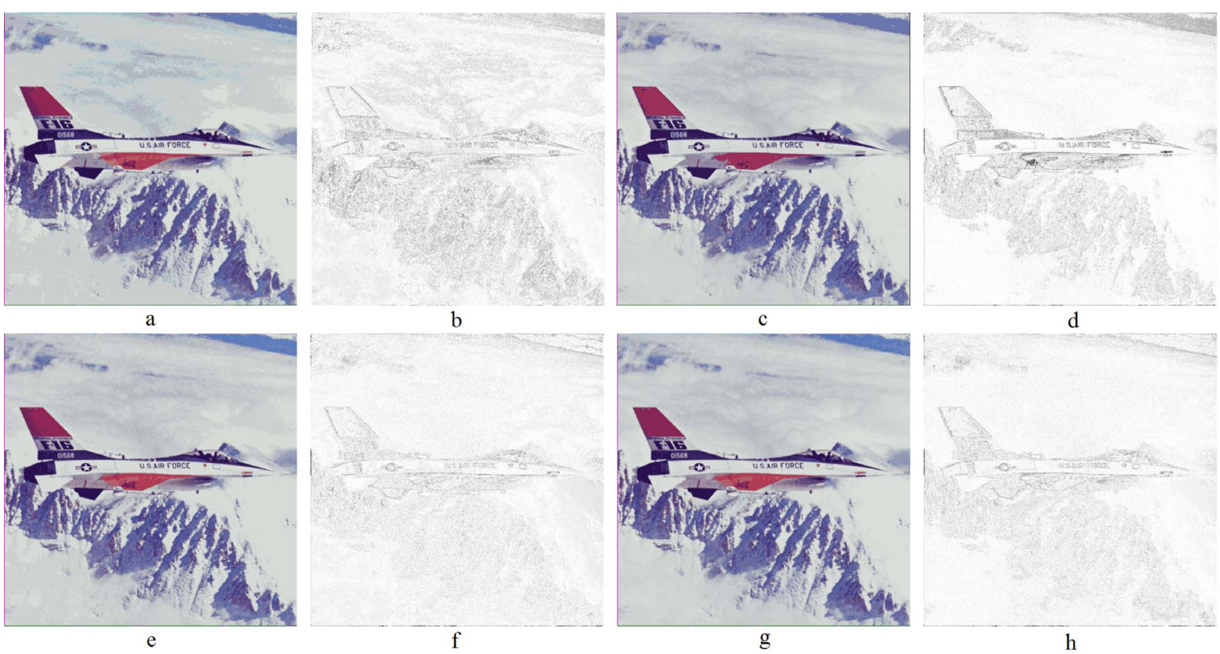


Fig. 2. (Color online) Sample quantization results for the Airplane image ($K=32$): a, MMM output; b, MMM error; c, NEU output; d, NEU error; e, WSM output; f, WSM error; g, WSM-C output; h, WSM-C error.

Fig. 3. (Color online) Sample quantization results for the Parrots image ($K=64$): a, MC output; b, MC error; c, FKM output; d, FKM error; e, WSM output; f, WSM error; g, WSM-C output; h, WSM-C error.

• In general, the fastest method is MC, which is followed by SAM, WAN, and WU. The slowest methods are KM-C, FCM, PIM, FKM-C, KM, and SKM.

• WSM-C is significantly faster than its convergent rivals. In particular, it provides up to 392 times speedup over KM-C with an average of 62.

• WSM is the fastest postclustering method. It provides up to 46 times speedup over KM with an average of 14.

• KM-C, FKM-C, and WSM-C are significantly more stable (particularly when $K$ is small) than their fixed-iteration counterparts as evidenced by their low standard deviation values in Table 1. This was expected, since these methods were allowed to run longer, which helped them overcome potentially adverse initial conditions.

Table 4 gives the mean stability ranks of the methods that involve random initialization. Given a test image and $K$ value combination, the stability of a method is calculated based on the coefficient of variation ($\sigma/\mu$) as $100(1-\sigma/\mu)$, where $\mu$ and $\sigma$ denote the mean and the standard deviation over 100 runs, respectively. Note that the $\mu$ and $\sigma$

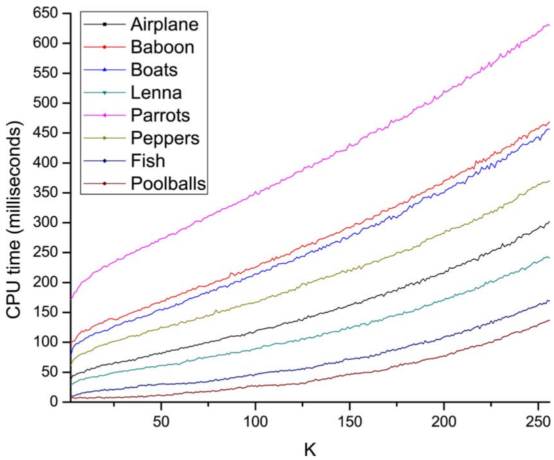


Fig. 4. (Color online) CPU time for WSM for $K=\{2,\ldots,256\}$.

values are given in Table 1. Clearly, the higher the stability of a method, the better. For example, when $K=32$, WSM-C obtains a mean MSE of 57.461492 with a standard deviation of 0.861126 on the Airplane image. Therefore, the stability of WSM-C in this case is calculated as $100(1-0.861126/57.461492)=98.50\%$. It can be seen that WSM-C is the most stable method, whereas WSM is the most stable fixed-iteration method.

Figures 2 and 3 show sample quantization results and the corresponding error images [49]. The error image for a particular quantization method was obtained by taking the pixelwise absolute difference between the original and quantized images. In order to obtain a better visualization, pixel values of the error images were multiplied by 4 and then negated. It can be seen that WSM-C and WSM obtain visually pleasing results with less prominent contouring. Furthermore, they achieve the highest color fidelity, which is evident by the clean error images that they produce.

Figure 4 illustrates the scaling behavior of WSM with respect to $K$. It can be seen that the complexity of WSM is sublinear in $K$, which is due to the intelligent use of the triangle inequality that avoids many distance computations once the cluster centers stabilize after a few iterations. For example, on the Parrots image, increasing $K$ from 2 to 256 results in an only about 3.67-fold increase in the computational time (172 versus 630 ms).

We should also mention two other KM-based quantization methods [24,27]. As in the case of FKM and SKM, these methods aim to accelerate KM without degrading its effectiveness. However, they do not address the stability problems of KM and thus provide almost the same results in terms of quality. In contrast, WSM (WSM-C) not only provides a considerable speedup over KM (KM-C), but also gives significantly better results, especially at lower quantization levels.

## 4. CONCLUSIONS

In this paper, a fast and effective color quantization method called WSM (weighted sort-means) was introduced. The method involves several modifications to the conventional $K$-means (KM) algorithm, including data reduction, sample weighting, and the use of the triangle inequality to speed up the nearest-neighbor search. Two variants of WSM were implemented. Although both have very reasonable computational requirements, the fixed-iteration variant is more appropriate for time-critical applications, while the convergent variant should be preferred in applications where obtaining the highest output quality is of prime importance, or the number of quantization levels or the number of unique colors in the original image is small. Experiments on a diverse set of images demonstrated that the two variants of WSM outperform state-of-the-art quantization methods with respect to distortion minimization. Future work will be directed toward the development of a more effective initialization method for WSM.

The implementation of WSM will be made publicly available as part of the Fourier image processing and analysis library, which can be downloaded from http://sourceforge.net/projects/fourier-ipal.

## ACKNOWLEDGMENTS

This publication was made possible by a grant from The Louisiana Board of Regents (LEQSF2008-11-RD-A-12). The author is grateful to Luiz Velho for the Fish image and Anthony Dekker for the Poolballs image.

## REFERENCES

1. L. Brun and A. Trémeau, *Digital Color Imaging Handbook* (CRC Press, 2002), pp. 589–638.
2. C.-K. Yang and W.-H. Tsai, "Color image compression using quantization, thresholding, and edge detection techniques all based on the moment-preserving principle," Pattern Recogn. Lett. **19**, 205–215 (1998).
3. Y. Deng and B. Manjunath, "Unsupervised segmentation of color–texture regions in images and video," IEEE Trans. Pattern Anal. Mach. Intell. **23**, 800–810 (2001).
4. O. Sertel, J. Kong, G. Lozanski, A. Shanaah, U. Catalyurek, J. Saltz, and M. Gurcan, "Texture classification using nonlinear color quantization: application to histopathological image analysis," in *IEEE International Conference on Acoustics, Speech and Signal Processing 2008, ICAASP 2008* (IEEE, 2008), pp. 597–600.
5. C.-T. Kuo and S.-C. Cheng, "Fusion of color edge detection and color quantization for color image watermarking using principal axes analysis," Pattern Recogn. **40**, 3691–3704 (2007).
6. Y. Deng, B. Manjunath, C. Kenney, M. Moore, and H. Shin, "An efficient color representation for image retrieval," IEEE Trans. Image Process. **10**, 140–147 (2001).
7. Z. Xiang, "Color quantization," in *Handbook of Approximation Algorithms and Metaheuristics* (Chapman & Hall/CRC, 2007), pp. 86-1–86-17.
8. R. S. Gentile, J. P. Allebach, and E. Walowit, "Quantization of color images based on uniform color spaces," J. Electron. Imaging **16**, 11–21 (1990).
9. P. Heckbert, "Color image quantization for frame buffer display," ACM SIGGRAPH Comput. Graph. **16**, 297–307 (1982).
10. M. Gervautz and W. Purgathofer, "A simple method for color quantization: octree quantization," in *New Trends in Computer Graphics* (Springer-Verlag, 1988), pp. 219–231.
11. S. Wan, P. Prusinkiewicz, and S. Wong, "Variance-based color image quantization for frame buffer display," Color Res. Appl. **15**, 52–58 (1990).
12. M. Orchard and C. Bouman, "Color quantization of images," IEEE Trans. Signal Process. **39**, 2677–2690 (1991).
13. X. Wu, "Efficient statistical computations for optimal color

quantization," in *Graphics Gems Volume II* (Academic, 1991), pp. 126–133.
14. G. Joy and Z. Xiang, "Center-cut for color image quantization," Visual Comput. **10**, 62–66 (1993).
15. C.-Y. Yang and J.-C. Lin, "RWM-cut for color image quantization," Comput. Graph. **20**, 577–588 (1996).
16. S. Cheng and C. Yang, "Fast and novel technique for color quantization using reduction of color space dimensionality," Pattern Recogn. Lett. **22**, 845–856 (2001).
17. Y. Sirisathitkul, S. Auwatanamongkol, and B. Uyyanonvara, "Color image quantization using distances between adjacent colors along the color axis with highest color variance," Pattern Recogn. Lett. **25**, 1025–1043 (2004).
18. K. Kanjanawanishkul and B. Uyyanonvara, "Novel fast color reduction algorithm for time-constrained applications," J. Visual Commun. Image Represent **16**, 311–332 (2005).
19. W. H. Equitz, "A new vector quantization clustering algorithm," IEEE Trans. Acoust., Speech, Signal Process. **37**, 1568–1575 (1989).
20. R. Balasubramanian and J. Allebach, "A new approach to palette selection for color images," J. Electron. Imaging **17**, 284–290 (1991).
21. Z. Xiang and G. Joy, "Color image quantization by agglomerative clustering," IEEE Comput. Graphics Appl. **14**, 44–48 (1994).
22. L. Velho, J. Gomez, and M. Sobreiro, "Color image quantization by pairwise clustering," in *X Brazilian Symposium on Computer Graphics and Image Processing 1977* (IEEE Computer Society, 1997), pp. 203–210.
23. L. Brun and M. Mokhtari, "Two high speed color quantization algorithms," in *Proceedings of the First International Conference on Color in Graphics and Image Processing* (IEEE, 2000), pp. 116–121.
24. H. Kasuga, H. Yamamoto, and M. Okamoto, "Color quantization using the fast *K*-means algorithm," Syst. Comput. Jpn. **31**, 33–40 (2000).
25. Y.-L. Huang and R.-F. Chang, "A fast finite-state algorithm for generating RGB palettes of color quantized images," J. Inf. Sci. Eng. **20**, 771–782 (2004).
26. Y.-C. Hu and M.-G. Lee, "*K*-means based color palette design scheme with the use of stable flags," J. Electron. Imaging **16**, 033003 (2007).
27. Y.-C. Hu and B.-H. Su, "Accelerated *K*-means clustering algorithm for colour image quantization," Imaging Sci. J. **56**, 29–40 (2008).
28. Z. Xiang, "Color image quantization by minimizing the maximum intercluster distance," ACM Trans. Graphics **16**, 260–276 (1997).
29. O. Verevka and J. Buchanan, "Local *K*-means algorithm for colour image quantization," in *Proceedings of the Graphics/Vision Interface Conference* (ACM, 1995), pp. 128–135.
30. P. Scheunders, "Comparison of clustering algorithms applied to color image quantization," Pattern Recogn. Lett. **18**, 1379–1384 (1997).
31. M. E. Celebi, "An effective color quantization method based on the competitive learning paradigm," in *Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition* (2009), pp. 876–880.
32. D. Ozdemir and L. Akarun, "Fuzzy algorithm for color quantization of images," Pattern Recogn. **35**, 1785–1791 (2002).
33. G. Schaefer and H. Zhou, "Fuzzy clustering for colour reduction in images," Telecommun. Syst. **40**, 17–25 (2009).
34. Z. Bing, S. Junyi, and P. Qinke, "An adjustable algorithm for color quantization," Pattern Recogn. Lett. **25**, 1787–1797 (2004).
35. A. Dekker, "Kohonen neural networks for optimal colour quantization," Network Comput. Neural Syst. **5**, 351–367 (1994).
36. N. Papamarkos, A. Atsalakis, and C. Strouthopoulos, "Adaptive color reduction," IEEE Trans. Syst., Man, Cybern., Part B: Cybern. **32**, 44–56 (2002).
37. C.-H. Chang, P. Xu, R. Xiao, and T. Srikanthan, "New adaptive color quantization method based on self-organizing maps," IEEE Trans. Neural Netw. **16**, 237–249 (2005).
38. Y. Linde, A. Buzo, and R. Gray, "An algorithm for vector quantizer design," IEEE Trans. Commun. **28**, 84–95 (1980).
39. G. Gan, C. Ma, and J. Wu, *Data Clustering: Theory, Algorithms, and Applications* (SIAM, 2007).
40. P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay, "Clustering large graphs via the singular value decomposition," Mach. Learn. **56**, 9–33 (2004).
41. S. Lloyd, "Least squares quantization in PCM," IEEE Trans. Inf. Theory **28**, 129–136 (1982).
42. E. Forgy, "Cluster analysis of multivariate data: efficiency vs. interpretability of classification," Biometrics **21**, 768 (1965).
43. S. Phillips, "Acceleration of *K*-means and related clustering algorithms," in *Proceedings of the 4th International Workshop on Algorithm Engineering and Experiments* (2002), pp. 166–177.
44. T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Wu, "An efficient *K*-means clustering algorithm: analysis and implementation," IEEE Trans. Pattern Anal. Mach. Intell. **24**, 881–892 (2002).
45. S. Har-Peled and A. Kushal, "Smaller coresets for *K*-median and *K*-means clustering," in *Proceedings of the 21st Annual Symposium on Computational Geometry* (2004), pp. 126–134.
46. C. Elkan, "Using the triangle inequality to accelerate *K*-means," in *Proceedings of the 20th International Conference on Machine Learning* (2003), pp. 147–153.
47. J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms* (Springer-Verlag, 1981).
48. J. F. Kolen and T. Hutcheson, "Reducing the time complexity of the fuzzy *C*-means algorithm," IEEE Trans. Fuzzy Syst. **10**, 263–267 (2002).
49. Uncompressed full-resolution images are available at http://www.lsus.edu/faculty/~ecelebi/color_quantization.htm.