# An Effective Color Quantization Method Based on the Competitive Learning Paradigm

**M. Emre Celebi**
Department of Computer Science
Louisiana State University, Shreveport, LA, USA
ecelebi@lsus.edu

**Abstract**— *Color quantization is an important operation with many applications in graphics and image processing. Most quantization methods are essentially based on data clustering algorithms one which is the popular k-means algorithm. A common drawback of many conventional clustering algorithms is the generation of empty clusters (dead units). In this paper, we apply Uchiyama and Arbib's competitive learning algorithm [1] to the problem of color quantization. In contrast to the conventional batch k-means algorithm, this competitive learning algorithm requires no cluster center initialization. In addition, it effectively avoids the dead unit problem by utilizing a simple cluster splitting rule. Experiments on commonly used test images demonstrate that the presented method outperforms various state-of-the-art methods in terms of quantization effectiveness.*

**Keywords:** Color quantization, clustering, k-means, competitive learning

## 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 [2]. Modern applications of color quantization include: (i) image compression, (ii) image segmentation, (iii) image analysis, (iv) image watermarking, and (v) content-based image retrieval.

The process of color quantization is mainly comprised 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 size palette is a difficult problem.

Color quantization methods can be broadly classified into two categories: image-independent methods that determine a universal (fixed) palette without regard to any specific image, 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 [2]. 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 [3], octree [4], variance-based method [5], binary splitting [6], greedy orthogonal bipartitioning [7], center-cut [8], and rwm-cut [9]. On the other hand, agglomerative preclustering methods [10], [11], [12], [13] 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 to preclustering methods at the expense of increased computational time. Clustering algorithms adapted to color quantization include k-means [14], [15], [16], [17], minmax [18], [19], competitive learning [20], [21], [22], fuzzy c-means [23], [24], BIRCH [25], and self-organizing maps [26], [27], [28], [29].

In this paper, we apply Uchiyama and Arbib's competitive learning algorithm [1] to the problem of color quantization. This algorithm requires no cluster center initialization and avoids the dead unit problem by using a simple cluster

splitting rule. The rest of the paper is organized as follows. Sections 2 and 3 describe the conventional batch k-means clustering algorithm and Uchiyama and Arbib's competitive learning algorithm, respectively. Section 4 presents the experimental results, whereas Section 5 gives the conclusions.

## 2. Batch K-Means Algorithm

The k-means algorithm [30] is inarguably one of the most widely used methods for data clustering [31], [32]. Given a data set $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \in \mathbb{R}^D$, the objective of k-means is to partition $X$ into $K$ exhaustive and mutually exclusive clusters $S = \{S_1, \ldots, S_k\}$, $\bigcup_{k=1}^{K} S_k = X$, $S_i \cap S_j \equiv \emptyset$ for $i \neq j$ by minimizing the sum of squared error (SSE):

$$\text{SSE} = \sum_{k=1}^{K} \sum_{\mathbf{x}_i \in S_k} \|\mathbf{x}_i - \mathbf{c}_k\|^2 \qquad (1)$$

where, $\|\,\|$ denotes the Euclidean norm and $\mathbf{c}_k$ is the center of cluster $S_k$ calculated as the mean (average) of points that belong to this cluster. This problem is known to be computationally intractable, but a heuristic method developed by Lloyd [33] offers a simple solution. Lloyd's algorithm starts with $K$ arbitrary centers, typically chosen uniformly at random from the data points. 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 Algo. (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 $\mathbf{x}_i$.

For the purpose of color quantization, the batch k-means algorithm has two main drawbacks. First, due to its batch nature, i.e. cluster centers are updated after the presentation of all the input vectors, the algorithm might get stuck in local optima. Second, the algorithm is quite sensitive to the initial choice of cluster centers. For example, a poor initialization scheme might lead to empty clusters (dead units).

**input** : $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \in \mathbb{R}^D$ ($N \times D$ input data set)
**output**: $C = \{\mathbf{c}_1, \ldots, \mathbf{c}_K\} \in \mathbb{R}^D$ ($K$ cluster centers)
Select a random subset $C$ of $X$ as the initial set of cluster centers;
**while** *termination criterion is not met* **do**
  **for** $(i = 1; i \leq N; i = i + 1)$ **do**
    Assign $\mathbf{x}_i$ to the nearest cluster;
    $m[i] = \underset{k \in \{1,\ldots,K\}}{\operatorname{argmin}} \|\mathbf{x}_i - \mathbf{c}_k\|^2$;
  **end**
  Recalculate the cluster centers;
  **for** $(k = 1; k \leq K; k = k + 1)$ **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 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**

**Algorithm 1**: Batch K-Means Algorithm

## 3. Adaptive Distributing Units Algorithm

The Adaptive Distributing Units (ADU) algorithm of Uchiyama and Arbib [1] is an online clustering algorithm based on the competitive learning paradigm, which is closely related to neural networks [32]. According to Rumelhart and Zipser [34], a competitive learning scheme consists of the following three basic components:

1) Start with a set of units that are all the same except for some randomly distributed parameter which makes each of them respond slightly differently to a set of input patterns.
2) Limit the 'strength' of each unit.
3) Allow the units to compete in some way for the right to respond to a given subset of inputs.

Due to their online nature, i.e. cluster centers are updated after the presentation of each input vector, clustering algorithms based on the competitive learning scheme are generally more adaptive. In other words, online clustering methods are more likely to escape local minima when compared to batch methods.

The pseudocode for the ADU algorithm is given in Algo. (2). Here $n$ and $wc[i]$ denote the number of cluster centers determined at a given time and the number of times that center $\mathbf{c}_i$ won the competition in the past, respectively. The algorithm parameters $\theta$, $t_{max}$, and $\gamma$ denote the maximum number of times a particular center can win, the maximum number of iterations, and the learning rate, respectively. The procedure starts with a single cluster whose center is given by the centroid of the input vectors. New cluster centers are added by splitting existing clusters that reach the threshold number of wins $\theta$ until the number of clusters reaches $K$. This splitting rule effectively avoids the dead unit problem. Following [1], the algorithm parameters were set to $\theta = 400\sqrt{K}$, $t_{max} = (2K - 3)\theta$, and $\gamma = 0.015$.

## 4. Experimental Results and Discussion

The ADU method was compared to some of the well-known quantization methods in the literature:

- **Median-cut (MC)** [3]: This method starts by building a $32 \times 32 \times 32$ color histogram that contains the

**input** : $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \in \mathbb{R}^D$ ($N \times D$ input data set)
**output**: $C = \{\mathbf{c}_1, \ldots, \mathbf{c}_K\} \in \mathbb{R}^D$ ($K$ cluster centers)
The first unit is given by the centroid of the input vectors;
$n = 1$;
$\mathbf{c}_n = 1/N \sum_{i=1}^{N} \mathbf{x}_i$;
Initialize the win counts to 0;
**for** $(i = 1; i \leq K; i = i + 1)$ **do**
  $wc[i] = 0$;
**end**
**for** $(t = 1; t \leq t_{max}; t = t + 1)$ **do**
  Select an input vector $\mathbf{x}$ randomly from $X$;
  Determine the winner (nearest) unit;
  $winner = \underset{k \in \{1,\ldots,n\}}{\operatorname{argmin}} \|\mathbf{x} - \mathbf{c}_k\|^2$;
  Update the winner and its win count;
  $\mathbf{c}_{winner} = \mathbf{c}_{winner} + \gamma (\mathbf{x} - \mathbf{c}_{winner})$;
  $wc[winner] = wc[winner] + 1$;
  Split the unit if its win count exceeds the threshold;
  **if** $wc[winner] = \theta$ && $n < K$ **then**
    $n = n + 1$;
    $\mathbf{c}_n = \mathbf{c}_{winner}$;
    $wc[n] = wc[winner] = 0$;
  **end**
**end**

**Algorithm 2**: Adaptive Distributing Units Algorithm

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)** [5]: 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)** [7]: 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)** [26]: This method utilizes a one-dimensional self-organizing map (Kohonen neural network) with 256 neurons. A pseudo-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 setting, i.e. $f = 1$, was used.
- **Local k-means (LKM)** [20]: This method uses hard competitive learning with an exponential decay to approximate an optimal palette using multiple subsets of image pixels.
- **Modified minmax (MMM)** [18]: 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 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.
- **Pairwise clustering (PWC)** [12]: This method is a modification of Ward's agglomerative hierarchical clustering method [35]. It builds a reduced color histogram and constructs a $Q \times Q$ joint quantization error matrix, where $Q$ represents the number colors in the reduced color histogram. The clustering procedure starts with $Q$ singleton clusters each of which contains one image pixel. At each iteration, the pair of clusters with the minimum joint quantization error are merged. This merging process is repeated until $K$ clusters remain. Note that this method has quadratic complexity in $Q$ and therefore the size of the reduced histogram was chosen as $16 \times 16 \times 16$ to limit the computational requirements.
- **Split & Merge (SAM)** [13]: 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 $B = 20K$.
- **Conventional k-means (KM)**: This method is implemented as outlined in Algo. (1) with random initialization [36] and a maximum number of iterations of 10.

The effectiveness of a quantization method was quantified by the commonly used Mean Squared Error (MSE) measure:

$$\text{MSE} = \frac{1}{HW} \sum_{h=1}^{H} \sum_{w=1}^{W} \left\| I(h,w) - \hat{I}(h,w) \right\|^2 \qquad (2)$$

where $I$ and $\hat{I}$ denote the original and the quantized images with height $H$ and width $W$, respectively. Note that the Peak Signal-to-Noise Ratio (PSNR) measure can be easily calculated from the MSE value:

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

Table 1 shows the performance of the methods at quantization levels 32, 64, 128, and 256 on some of the most commonly used test images in the literature, namely Baboon, Lenna, Parrots, and Peppers. The best (lowest) MSE values are shown in **bold**. Note that since KM and ADU involve randomness, the quantization errors for these methods are specified in the form of mean/standard deviation over 50 runs. It can be seen that the most effective method is ADU, which is followed by KM, NEU, and SAM. In addition, ADU is significantly more stable when compared to KM despite the fact that both of them are nondeterministic.

Figure 1 shows the quantization results for the Parrots image and the corresponding error images. The error image associated with a particular quantization method was obtained by taking the pixelwise absolute difference between the original and the 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 ADU obtains visually pleasing results with less prominent contouring at low quantization levels. Furthermore, it achieves the highest color fidelity which is evident by its clean error image.

# 5. Conclusions

In this paper, an effective color quantization method based on the competitive learning paradigm was introduced. The method is advantageous over conventional clustering methods in that it requires no cluster center initialization and that it avoids the dead unit problem. Experiments on a diverse set of images demonstrated that the presented method outperforms state-of-the-art methods in terms of quantization effectiveness.

The implementation of the presented color quantization method 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).

# References

[1] T. Uchiyama and M. Arbib, "An Algorithm for Competitive Learning in Clustering Problems," *Pattern Recognition*, vol. 27, no. 10, pp. 1415–1421, 1994.

[2] L. Brun and A. Trémeau, *Digital Color Imaging Handbook*. CRC Press, 2002, ch. Color Quantization, pp. 589–638.

[3] P. Heckbert, "Color Image Quantization for Frame Buffer Display," *ACM SIGGRAPH Computer Graphics*, vol. 16, no. 3, pp. 297–307, 1982.

[4] M. Gervautz and W. Purgathofer, *New Trends in Computer Graphics*. Springer-Verlag, 1988, ch. A Simple Method for Color Quantization: Octree Quantization, pp. 219–231.

[5] S. Wan, P. Prusinkiewicz, and S. Wong, "Variance-Based Color Image Quantization for Frame Buffer Display," *Color Research and Application*, vol. 15, no. 1, pp. 52–58, 1990.

[6] M. Orchard and C. Bouman, "Color Quantization of Images," *IEEE Transactions on Signal Processing*, vol. 39, no. 12, pp. 2677–2690, 1991.

[7] X. Wu, *Graphics Gems Volume II*. Academic Press, 1991, ch. Efficient Statistical Computations for Optimal Color Quantization, pp. 126–133.

[8] G. Joy and Z. Xiang, "Center-Cut for Color Image Quantization," *The Visual Computer*, vol. 10, no. 1, pp. 62–66, 1993.

[9] C.-Y. Yang and J.-C. Lin, "RWM-Cut for Color Image Quantization," *Computers and Graphics*, vol. 20, no. 4, pp. 577–588, 1996.

[10] R. Balasubramanian and J. Allebach, "A New Approach to Palette Selection for Color Images," *Journal of Imaging Technology*, vol. 17, no. 6, pp. 284–290, 1991.

[11] Z. Xiang and G. Joy, "Color Image Quantization by Agglomerative Clustering," *IEEE Computer Graphics and Applications*, vol. 14, no. 3, pp. 44–48, 1994.

[12] L. Velho, J. Gomez, and M. Sobreiro, "Color Image Quantization by Pairwise Clustering," in *Proceedings of the Brazilian Symposium of Computer Graphics and Image Processing*, 1997, pp. 203–210.

[13] L. Brun and M. Mokhtari, "Two High Speed Color Quantization Algorithms," in *Proceedings of the 1st International Conference on Color in Graphics and Image Processing*, 2000, pp. 116–121.

[14] H. Kasuga, H. Yamamoto, and M. Okamoto, "Color Quantization Using the Fast K-Means Algorithm," *Systems and Computers in Japan*, vol. 31, no. 8, pp. 33–40, 2000.

[15] Y.-L. Huang and R.-F. Chang, "A Fast Finite-State algorithm for Generating RGB Palettes of Color Quantized Images," *Journal of Information Science and Engineering*, vol. 20, no. 4, pp. 771–782, 2004.

[16] Y.-C. Hu and M.-G. Lee, "K-means Based Color Palette Design Scheme with the Use of Stable Flags," *Journal of Electronic Imaging*, vol. 16, no. 3, p. 033003, 2007.

[17] Y.-C. Hu and B.-H. Su, "Accelerated K-means Clustering Algorithm for Colour Image Quantization," *Imaging Science Journal*, vol. 56, no. 1, pp. 29–40, 2008.

[18] Z. Xiang, "Color Image Quantization by Minimizing the Maximum Intercluster Distance," *ACM Transactions on Graphics*, vol. 16, no. 3, pp. 260–276, 1997.

[19] I.-S. Hsieh and K.-C. Fan, "An Adaptive Clustering Algorithm for Color Quantization," *Pattern Recognition Letters*, vol. 21, no. 4, pp. 337–346, 2000.

[20] O. Verevka and J. Buchanan, "Local K-Means Algorithm for Colour Image Quantization," in *Proceedings of the Graphics/Vision Interface Conference*, 1995, pp. 128–135.

[21] P. Scheunders, "Comparison of Clustering Algorithms Applied to Color Image Quantization," *Pattern Recognition Letters*, vol. 18, no. 11-13, pp. 1379–1384, 1997.

[22] L. Law and Y. Cheung, "Color Image Segmentation Using Rival Penalized Controlled Competitive Learning," in *Proceedings of the 2003 International Joint Conference on Neural Networks*, 2003, pp. 108–112.

[23] D. Ozdemir and L. Akarun, "Fuzzy Algorithm for Color Quantization of Images," *Pattern Recognition*, vol. 35, no. 8, pp. 1785–1791, 2002.

[24] G. Schaefer and H. Zhou, "Fuzzy Clustering for Colour Reduction in Images," *Telecommunication Systems*, vol. 40, no. 1-2, pp. 17–25, 2009.

[25] Z. Bing, S. Junyi, and P. Qinke, "An Adjustable Algorithm for Color Quantization," *Pattern Recognition Letters*, vol. 25, no. 16, pp. 1787–1797, 2004.

[26] A. Dekker, "Kohonen Neural Networks for Optimal Colour Quantization," *Network: Computation in Neural Systems*, vol. 5, no. 3, pp. 351–367, 1994.

[27] N. Papamarkos, A. Atsalakis, and C. Strouthopoulos, "Adaptive Color Reduction," *IEEE Transactions on Systems, Man, and Cybernetics Part B*, vol. 32, no. 1, pp. 44–56, 2002.

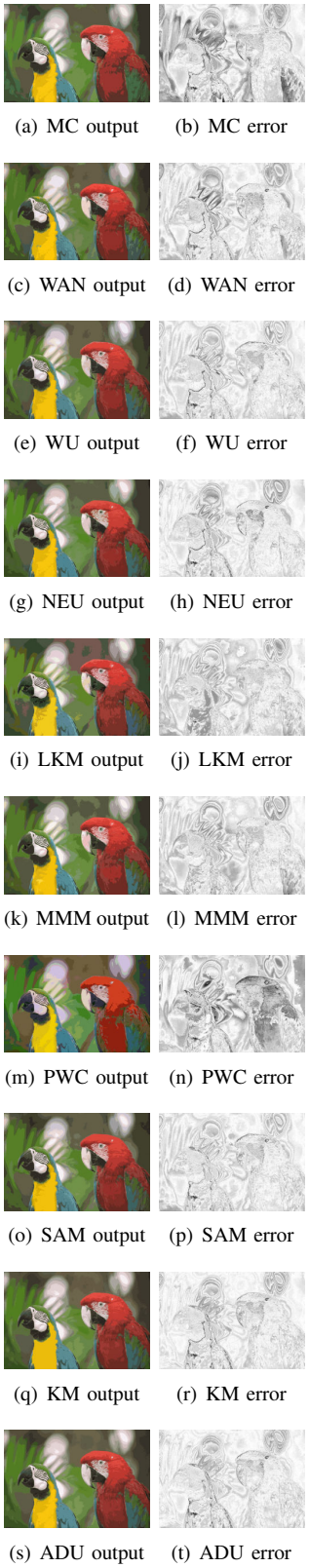
[28] C.-H. Chang, P. Xu, R. Xiao, and T. Srikanthan, "New Adaptive Color Quantization Method Based on Self-Organizing Maps," *IEEE Transactions on Neural Networks*, vol. 16, no. 1, pp. 237–249, 2005.

[29] C.-H. Wang, C.-N. Lee, and C.-H. Hsieh, "Sample-Size Adaptive Self-Organization Map for Color Images Quantization," *Pattern Recognition Letters*, vol. 28, no. 13, pp. 1616–1629, 2007.

[30] Y. Linde, A. Buzo, and R. Gray, "An Algorithm for Vector Quantizer Design," *IEEE Transactions on Communications*, vol. 28, no. 1, pp. 84–95, 1980.
[31] G. Gan, C. Ma, and J. Wu, *Data Clustering: Theory, Algorithms, and Applications*. SIAM, 2007.
[32] R. Xu and D. Wunsch, *Clustering*. Wiley-IEEE Press, 2008.
[33] S. Lloyd, "Least Squares Quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–136, 1982.
[34] D. Rumelhart and D. Zipser, "Feature Discovery by Competitive Learning," *Cognitive Science: A Multidisciplinary Journal*, vol. 9, no. 1, pp. 75–112, 1985.
[35] J. Ward, "Hierarchical Grouping to Optimize an Objective Function," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 236–244, 1963.
[36] E. Forgy, "Cluster Analysis of Multivariate Data: Efficiency vs. Interpretability of Classification," *Biometrics*, vol. 21, p. 768, 1965.

Table 1
MSE COMPARISON OF THE QUANTIZATION METHODS

| | Baboon (512×512) | | | | Lenna (512 × 480) | | | |
|---|---|---|---|---|---|---|---|---|
| | K | | | | K | | | |
| Method | 32 | 64 | 128 | 256 | 32 | 64 | 128 | 256 |
| MC | 546 | 371 | 248 | 166 | 165 | 94 | 71 | 47 |
| WAN | 509 | 326 | 216 | 142 | 159 | 93 | 61 | 43 |
| WU | 422 | 248 | 155 | 99 | 130 | 76 | 46 | 29 |
| NEU | 363 | 216 | 128 | 84 | 119 | 68 | 36 | 23 |
| LKM | 407 | 216 | 145 | 92 | 123 | 73 | 42 | 33 |
| MMM | 489 | 270 | 189 | 120 | 139 | 86 | 50 | 34 |
| PWC | 422 | 254 | 161 | 104 | 164 | 103 | 72 | 63 |
| SAM | 396 | 245 | 153 | 99 | 135 | 88 | 56 | 40 |
| KM | 346/12 | 204/3 | 129/2 | 83/1 | 107/7 | 61/2 | 37/1 | 24/0 |
| ADU | **332**/3 | **197**/1 | **124**/0 | **78**/0 | **98**/1 | **57**/0 | **34**/0 | **22**/0 |
| | Parrots (1536 × 1024) | | | | Peppers (512 × 512) | | | |
| | K | | | | K | | | |
| Method | 32 | 64 | 128 | 256 | 32 | 64 | 128 | 256 |
| 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 |
| LKM | 406 | 185 | 113 | 59 | 286 | 173 | 100 | 62 |
| MMM | 332 | 200 | 117 | 73 | 292 | 182 | 113 | 76 |
| PWC | 745 | 489 | 340 | 274 | 286 | 188 | 117 | 80 |
| SAM | 276 | 160 | 94 | 60 | 268 | 161 | 100 | 64 |
| KM | 263/16 | 154/11 | 86/4 | 51/2 | 231/6 | 142/5 | 87/2 | 55/1 |
| ADU | **239**/4 | **133**/2 | **76**/1 | **45**/0 | **223**/2 | **133**/1 | **81**/0 | **50**/0 |



(a) MC output (b) MC error
(c) WAN output (d) WAN error
(e) WU output (f) WU error
(g) NEU output (h) NEU error
(i) LKM output (j) LKM error
(k) MMM output (l) MMM error
(m) PWC output (n) PWC error
(o) SAM output (p) SAM error
(q) KM output (r) KM error
(s) ADU output (t) ADU error

Fig. 1
QUANTIZATION RESULTS FOR PARROTS ($K = 32$)