J. Vis. Commun. Image R. 78 (2021) 103180

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

journal homepage: www.elsevier.com/locate/jvci

Full Length Article

Revisiting the Iterative Ant-tree for color quantization algorithm ☆

María-Luisa Pérez-Delgado

University of Salamanca, Escuela Politécnica Superior de Zamora, Av. Requejo, 33, C.P. 49022, Zamora, Spain

ARTICLE INFO

Keywords:
Color quantization
Clustering
Artificial ants

ABSTRACT

The Iterative Ant-tree for color quantization algorithm has recently been proposed to reduce the colors of an image at a low computational cost. It is a clustering-based method that generates good images compared to several well-known color quantization methods. This article proposes the modification of two features of the original algorithm: the value assigned to the parameter associated with the algorithm and the order in which the pixels of the image are processed. As a result, the new variant of the algorithm generates better images than the original and the results are less sensitive to the value selected for the parameter.

1. Introduction

A color image represented in the RGB color space is defined by a set of pixels distributed in a rows and b columns. Each pixel p_{ij}, with $1 \leq i \leq a$ and $1 \leq j \leq b$ is represented by three values between 0 and 255 that correspond to the amount of red, green and blue of said pixel: $p_{ij} = (R_{ij}, G_{ij}, B_{ij})$.

Given a color image, the objective of a color quantization method is to obtain another image with the same number of pixels as the original but represented with fewer colors. For this purpose, the color quantization methods define a quantized palette with a number of colors q much smaller than the size of the original palette. After this, the quantized image is obtained by representing each pixel of the original image by a color of the quantized palette, in such a way that the new image is as similar as possible to the original one.

Reducing the number of different colors of an image allows it to be displayed on low-end devices. In addition, this also reduces the size of the image, which improves the transmission speed of the image and optimizes the storage space.

Garey et al. showed that finding the optimal quantized palette is an NP-complete problem [1]. For this reason, researchers have proposed several heuristic methods for solving the color quantization problem. The Iterative Ant-tree for color quantization (ITATCQ) algorithm is a recently proposed method of this type [2]. It is based on the Ant-tree for color quantization (ATCQ) algorithm [3], which represents the pixels of the image by artificial ants and organizes them in a tree structure to define the quantized palette and to generate the quantized image. The number of subtrees included in the tree defines the number of colors of the quantized palette. This value is conditioned by the features of the original image and also by the parameter α of the algorithm. Said parameter is used to compute a threshold that determines when an ant is connected to a certain subtree and when a new subtree must be defined. Computational results reported in [3] show that the value of the α parameter influences the quality of the quantized image.

The ITATCQ algorithm is an iterative method in which each iteration applies the operations of the ATCQ algorithm. The first iteration builds an initial tree which defines an initial quantized palette; then, the following iterations improve said palette. Each of these iterations first disconnects all the ants from the tree and then applies the ATCQ operations to reconnect them. Although ATCQ works on data sorted by average similarity, ITATCQ does not consider this sorting and the pixels are taken sequentially from the original image. In any case, the α parameter also influences the quality of the final image [2].

This article proposes two modifications to ITATCQ: the use of an α value that is updated along the iterations and the random selection of the pixels to be processed. Computational experiments show that both modifications improve the results of the algorithm and reduce the influence of the value selected for the α parameter.

The rest of the article is organized as follows. Section 2 gives an overview of the color quantization methods proposed in the literature. Next, Section 3 describes the operations of the ITATCQ algorithm, while Section 4 presents the modifications proposed in this article for this algorithm. Said modifications were applied to several images and the results are discussed in Section 5, which also includes the comparison with other color quantization methods. Finally, the article includes the conclusions in Section 6.

2. Color quantization methods

The methods proposed to solve the color quantization problem can be divided into two categories: splitting methods and clustering-based

☆ This paper has been recommended for acceptance by Zicheng Liu.

E-mail address: mlperez@usal.es.

https://doi.org/10.1016/j.jvcir.2021.103180
Received 17 June 2019; Received in revised form 31 January 2020; Accepted 3 June 2021
Available online 9 June 2021

methods. In general, the methods of the first category are faster, but the methods of the second one can generate better quantized images.

The splitting methods recursively divide the color space until q regions are defined. Then, they generate a quantized palette that includes a color to represent each region. These methods differ with regard to two factors: the region selected to perform the next division and the splitting point considered. Some popular methods of this type are Median-cut [4], Variance-based method [5], Binary splitting [6], Octree [7], and Wu's methods [8,9].

The Median-cut method selects the box with more pixels and splits it along the longest axis at the median point. The centroids of the boxes obtained by the splitting process are used to define the quantized palette.

The Variance-based method splits the box with the largest weighted variance along the axis with the least weighted sum of projected variances. This method uses the point that minimizes the marginal squared error as the splitting point.

The Binary splitting method takes the box with the largest dominant eigenvalue and splits it along the principal axis of the box that is being split. In this case the splitting point is the projection of the centroid to the selected axis.

The Octree method builds a tree which can include 8 children per node as a maximum. The first step of this method builds a tree that represents the pixels of the original image. The second step progressively merges the leaves that represent fewer pixels, until the tree only includes q leaves.

The Greedy orthogonal bipartitioning method is based on the same idea as the Variance-based method, but the splitting axis is the one that minimizes the sum of the variances of both sides [8]. The method proposed in [9] sorts the colors of the image along their principal axis and applies dynamic programming to divide the color space based on this ordering.

In addition to the splitting methods described, some other methods are presented in [10–14].

Clustering-based methods define groups or clusters with similar pixels and represent all the pixels in the same cluster by the same color. Some clustering methods that have been applied to solve the color quantization problem are K-means, artificial neural networks or swarm-based algorithms.

K-means is a well-known clustering method that selects a set of q initial centroids and then associates each point with the closest centroid. After this, the centroids are recalculated from the points associated with each of the q clusters. The operations are applied iteratively until a predefined number of iterations has been performed or until a predetermined error has been reached. This method has been applied to color quantization in several articles [15–18]

The Neuquant method applies a self-organizing feature map to solve the color quantization problem [19]. This method uses the pixels of the image to train a network with q neurons whose final weights define the colors of the quantized palette. Other solutions based on the use of neural networks are presented in [20–24].

Omran et al. [25] proposed a color quantization method that combines the Particle swarm optimization algorithm [26] with K-means. This proposal applies several iterations of K-means to each particle in order to improve the solution represented by said particle.

Ozturk et al. [27] combined the Artificial bee colony (ABC) algorithm [28] with K-means in a similar way to that proposed by Omran et al. [25].

The ATCQ algorithm [3] adapts the Ant-tree algorithm [29] to perform color quantization. In this case, the pixels of the image are represented by ants that are connected in a tree structure taking into account the similarity of the colors corresponding to the ants. The pixels of each subtree define a cluster and are represented in the quantized image by the color associated with that subtree.

In addition to ITATCQ, which is described in detail in the following section, several color quantization methods based on ATCQ have recently been proposed. ATCQ was combined with the Firefly algorithm [30] in the ATCQ+FA algorithm [31]. In this case, ATCQ was applied to compute the fitness of each firefly and to improve the solution represented by it. The ABC+ATCQ method combines the ABC algorithm with ATCQ [32], generating a faster algorithm than the one proposed by Ozturk et al. [27]. This method uses the tree defined by ATCQ to compute the fitness of a food source. This information is used to decide if a candidate source should replace another source that is currently being exploited by a bee. The BS+ATCQ method applies the ATCQ operations to the binary tree generated by the Binary splitting algorithm, improving the results obtained when each of the methods is applied separately [33].

The Shuffled-frog leaping algorithm (SFLA) was applied to generate a quantized image in [34]. In this case, each frog represents a quantized palette and the algorithm is applied to a subset of pixels of the original image in order to reduce the execution time.

Some other clustering-based methods have been applied to color quantization in [35–40].

3. The ITATCQ algorithm

The algorithm associates an ant h_{ij} to each pixel p_{ij} of the original image and organizes these ants in a tree structure with three levels. The first level includes the root, called support and denoted by a_0. The second level includes the children of a_0, denoted by $S_1, S_2, \ldots, S_q$, which are the roots of the q subtrees of ants created by the algorithm. The third level includes the ants that are progressively connected to the structure.

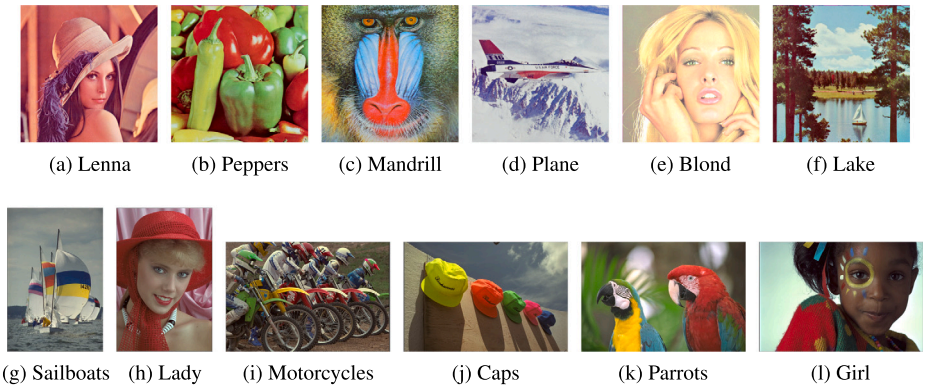
Algorithm 1 ITATCQ

```
Set q = 0
for iter = 1 to IT do
  for each ant, h_ij, in the support do
    if (q = 0) then
      Add_Subtree(tree, q, h_ij)
    else
      Select the subtree c with the color most similar to h_ij. Let
      d_ij_c =Sim(h_ij, color(c))
      if (d_ij_c < T) and (q < Q_max) then
        Add_Subtree(tree, q, h_ij)
      else
        Modify_Subtree(tree, c, h_ij, d_ij_c)
      end if
    end if
  end for
  if ( iter ≠ IT) then
    for c = 1 to q do
      Move all the children of S_c to the support
    end for
  end if
end for
```

The number of children of the support determines the number of colors of the quantized palette. For this reason, the number of children of a_0 is limited to Q_{max} ($q \leq Q_{max}$).

Each node S_c of the second level stores three values: $nc(c)$, $sum(c)$ and $e(c)$. The value $nc(c)$ represents the number of ants connected to the subtree c while $sum(c)$ represents the sum of the colors of such ants. These variables are used to compute the color of the subtree c: $color(c) = sum(c)/nc(c)$. On the other hand, $e(c)$ is computed by Eq. (1), where $d_{ij\ c}$ is the similarity between the color of the subtree c and the ant h_{ij} when such ant is included in said subtree.

$$e(c) = \sum_{h_{ij} \in c} d_{ij\ c} \tag{1}$$



(a) Lenna (b) Peppers (c) Mandrill (d) Plane (e) Blond (f) Lake

(g) Sailboats (h) Lady (i) Motorcycles (j) Caps (k) Parrots (l) Girl

Fig. 1. Images used for the tests. Size of the images: 512 × 512 for images (a) to (f), 512 × 768 for images (g) and (h), 768 × 512 for images (i) to (l).

The operations of ITATCQ that build the tree of ants are summarized in Algorithm 1. The ants are initially stored in a list that represents the pixels of the original image in raster order, that is, from left to right and from top to bottom. Each iteration of the algorithm processes all the ants in the list sequentially.

Initially, the tree only includes the node a_0 and all the ants are on it. Next, a total of IT iterations are applied, each including two main operations: the first operation connects all the ants to the tree and the second one moves all the ants back to the support. The last iteration does not apply the second operation because the final tree is needed to define the quantized palette and the quantized image. When the iterations conclude, the colors of the q nodes of the second level define the quantized palette. To define the quantized image, all the ants (pixels) that belong to the subtree c are represented by the color of said subtree.

At each iteration, all the ants are analyzed and connected to the tree. Let h_{ij} denote the ant selected to be analyzed. If the tree does not include any subtree ($q = 0$), the first one is created and h_{ij} connects to it. Otherwise, if $q > 0$, the algorithm selects the subtree c with the color most similar to that of h_{ij}; let $d_{ij\ c}$ denote said similarity. The similarity is compared to a threshold T given by Eq. (2), where $\alpha \in (0, 1]$. If $d_{ij\ c}$ does not reach T and a_0 does not include the maximum number of children allowed, a new subtree is created and h_{ij} connects to it. Otherwise, h_{ij} moves to the subtree c.

$$T = \frac{e(c)}{nc(c)}\alpha \tag{2}$$

When the ant h_{ij} must connect to a new subtree, first the new child of a_0, S_q, is created and initial values are given to the parameters of this node (h_{ij} defines the initial value of $sum(q)$, $nc(q)$ is set to 1 and $e(q)$ is set to 0). Next, h_{ij} becomes the first child of S_q (Algorithm 2).

Algorithm 2 Add_Subtree

Require: $tree$, q, h_{ij}
1: Set $q = q + 1$
2: Add the child S_q of the support
3: Set $sum(q) = (R_{ij}, G_{ij}, B_{ij})$
4: Set $e(q) = 0$, $nc(q) = 1$
5: Connect ant h_{ij} to S_q

When the ant h_{ij} must connect to an existing subtree, the values associated with the root of that subtree must be updated: $nc(c)$ increases by 1 because a new ant has been connected to the subtree, the similarity between the subtree and the ant is added to $e(c)$ and the color of the ant is added to $sum(c)$ (Algorithm 3).

The final operation of each iteration, except for the last one, disconnects all the ants from the tree and moves them back to the support. Therefore, starting from the second iteration, the tree already includes nodes in the second level. Successive iterations update the colors associated with the nodes of the second level and can also add more nodes to this level (up to Q_{max}). In this way, the iterative process updates the colors of the quantized palette. Computational results show that the quantized image improves as the iterations progress [2].

Algorithm 3 Modify_Subtree

Require: $tree$, c, h_{ij}, $d_{ij\ c}$
1: Set $sum(c) = sum(c) + (R_{ij}, G_{ij}, B_{ij})$
2: Set $e(c) = e(c) + d_{ij\ c}$, $nc(c) = nc(c) + 1$
3: Connect ant h_{ij} to S_c

4. The ITATCQ improvement

This article proposes to improve the ITATCQ results by modifying two features of the algorithm:

- the value of the α parameter.
- the procedure for selecting the pixels processed by the algorithm.

4.1. The α parameter

The α parameter greatly influences the ITATCQ results [2]. When this parameter is too small, the quantized palette does not include Q_{max} colors, so the quality of the quantized image is poor. On the other hand, when this parameter is too large, the quality of the image is also bad, although it may include Q_{max} different colors. As proposed in [2], a reasonable selection for the α parameter is the smallest value that generates a quantized palette that includes Q_{max} colors. In any case, it should be taken into account that the features of the analyzed image influence the selection, and the result is not the same for different images processed with the same α value and the same palette size. This makes it necessary to test several values of the parameter for a given image and then to make a choice.

The original ITATCQ uses the same α value during all the iterations. This article proposes the use of a value that varies over the course of the iterations. As described above, α is used to compute the threshold that determines when a pixel is associated with a color of the quantized palette and also when a new color of the palette must be created. When α is large, the Q_{max} colors of the quantized palette are defined during the initial iterations of the algorithm. Although this result may seem interesting, computational experiments show that the final image has lower quality than other images obtained with smaller α values that require more iterations to complete the colors of the quantized palette. For this reason, the proposed strategy considers a small initial value of α and then increases it as the iterations proceed.

Computational results reported in Section 5 show that the quantized image is better when α increases along the iterations than when this parameter is constant.

Table 1
MSE (*min*: minimum, *av*: average, *dev*: standard deviation) and average time (milliseconds) for ITATCQ-n at iteration 10.

		32 colors				64 colors				128 colors				256 colors			
	α	*min*	*av*	*dev*	*T*	*min*	*av*	*dev*	*T*	*min*	*av*	*dev*	*T*	*min*	*av*	*dev*	*T*
Lenna	0.25	121.82	127.44	5.2	825	74.55	76.01	1.0	1186	47.75	48.68	0.5	1870	31.91	32.23	0.3	3245
	0.30	140.89	155.27	16.2	844	76.35	78.13	1.0	1193	48.16	49.20	0.5	1868	31.61	32.08	0.3	3270
	0.35	125.07	149.11	31.2	875	79.67	84.91	8.4	1234	49.69	50.16	0.4	1944	31.81	32.09	0.2	3363
	0.40	123.65	133.72	8.0	860	75.94	78.57	1.9	1254	48.57	49.90	0.6	1975	32.28	32.54	0.2	3423
	0.45	121.26	127.70	5.8	866	73.80	75.66	1.0	1259	47.61	48.92	0.5	1992	32.23	32.72	0.3	3514
	0.50	120.52	124.96	3.9	857	75.00	75.84	0.5	1249	47.50	48.45	0.6	1986	31.57	32.07	0.3	3529
			136.37		**854**		**78.19**		**1229**		**49.22**		**1939**		**32.29**		**3391**
Peppers	0.25	235.57	261.33	18.5	848	141.63	152.01	7.1	1180	89.90	91.70	1.4	1873	57.72	59.11	1.1	3263
	0.30	241.88	257.50	16.0	831	143.28	150.64	3.5	1177	88.33	91.54	2.7	1870	57.68	58.30	0.5	3254
	0.35	242.55	254.65	15.5	836	140.10	160.37	14.8	1209	87.28	92.75	3.9	1893	56.91	58.34	1.1	3241
	0.40	237.27	262.77	20.2	861	148.73	160.95	5.9	1256	93.35	99.05	4.1	2007	59.35	60.42	0.9	3368
	0.45	235.59	259.64	18.6	849	144.77	157.15	5.9	1272	90.88	96.35	4.0	2076	59.25	60.64	0.7	3481
	0.50	244.74	256.49	8.5	867	138.64	148.36	5.7	1243	88.66	92.64	3.4	2040	57.20	58.50	0.7	3529
			258.73		**849**		**154.91**		**1223**		**94.00**		**1960**		**59.22**		**3356**
Mandrill	0.25	390.41	406.35	11.5	828	245.52	254.26	5.6	1176	160.11	163.00	2.3	1855	105.24	106.21	0.9	3234
	0.30	382.96	399.22	17.5	830	243.65	252.90	5.9	1192	156.99	161.31	2.7	1828	102.31	104.93	1.3	3212
	0.35	401.34	437.39	37.1	870	255.54	274.80	12.4	1235	161.99	164.86	2.2	1927	103.63	105.57	1.0	3268
	0.40	390.07	417.24	19.9	873	252.91	260.94	7.2	1269	162.95	166.12	2.0	1968	104.15	106.21	1.4	3366
	0.45	387.87	401.75	17.4	876	247.48	256.38	9.5	1241	159.99	165.06	2.9	2019	104.13	107.03	1.6	3581
	0.50	385.15	404.19	11.9	868	243.57	250.35	4.4	1235	158.68	163.22	2.4	2022	103.69	105.25	1.1	3593
			411.02		**857**		**258.27**		**1225**		**163.93**		**1937**		**105.87**		**3375**
Plane	0.25	75.75	91.03	11.0	826	41.42	44.57	2.0	1193	25.67	27.17	0.8	1942	16.32	16.76	0.3	3215
	0.30	85.10	100.36	9.7	849	43.34	50.10	3.6	1222	26.23	27.48	0.6	1959	16.04	16.62	0.4	3214
	0.35	74.21	89.81	9.3	858	47.92	54.18	4.2	1237	26.19	27.84	1.3	1958	16.05	16.57	0.3	3236
	0.40	68.98	76.92	4.5	844	44.02	46.50	1.7	1261	26.74	29.76	2.4	2067	16.14	16.64	0.3	3300
	0.45	70.25	75.16	4.5	849	40.28	44.05	2.4	1240	25.17	27.13	0.7	2033	16.80	17.12	0.3	3353
	0.50	66.11	74.04	6.0	844	38.80	42.28	1.6	1223	25.53	26.98	0.9	2050	16.13	16.64	0.3	3431
			84.55		**845**		**46.95**		**1229**		**27.73**		**2001**		**16.73**		**3292**
Blond	0.25	92.58	104.30	7.8	803	53.47	57.70	3.9	1193	32.48	33.69	1.0	1911	21.28	22.17	0.5	3241
	0.30	92.32	102.00	5.2	832	55.18	59.36	3.2	1212	32.62	34.07	1.1	1914	20.36	20.99	0.3	3321
	0.35	89.92	98.65	4.3	840	57.25	61.11	2.7	1258	32.96	35.50	1.5	1966	20.21	20.77	0.4	3304
	0.40	90.35	97.29	5.2	849	53.47	56.73	2.0	1242	35.31	38.11	2.1	1977	19.90	20.86	0.5	3344
	0.45	85.17	90.87	3.8	834	51.75	54.28	1.8	1279	33.41	34.53	0.9	1981	21.48	22.52	0.7	3426
	0.50	84.48	92.24	6.6	840	51.69	53.69	1.7	1240	31.54	32.94	0.7	1965	20.83	21.78	0.7	3433
			97.56		**833**		**57.14**		**1237**		**34.81**		**1952**		**21.52**		**3345**
Lake	0.25	208.71	224.35	17.2	799	138.35	143.48	3.3	1195	90.44	92.26	1.4	1883	57.83	59.06	0.7	3274
	0.30	211.55	227.77	17.9	846	137.45	142.68	2.8	1207	89.90	92.56	1.2	1908	58.58	59.48	0.8	3253
	0.35	211.26	227.61	15.5	828	136.85	143.87	5.2	1201	89.63	92.19	2.4	1901	57.48	58.96	0.9	3351
	0.40	212.41	220.43	7.6	819	135.47	145.05	4.0	1194	89.50	92.53	2.1	1931	58.06	59.84	1.2	3311
	0.45	209.39	219.35	8.4	829	140.62	143.71	2.4	1248	90.11	92.18	1.5	1973	59.43	60.42	0.8	3471
	0.50	207.30	222.98	19.4	846	138.42	143.65	4.0	1232	88.62	91.19	1.4	1983	58.35	59.46	0.6	3480
			223.75		**828**		**143.74**		**1213**		**92.15**		**1930**		**59.54**		**3357**

4.2. The processing order of the pixels

ITATCQ is an iterative variant of the ATCQ algorithm. On the other hand, ATCQ adapts the Ant-tree algorithm [29] for color quantization purposes. Since Ant-tree is applied to sorted data, ATCQ also does so, considering the pixels of the image sorted by increasing average similarity.

When ITATCQ was proposed, the use of sorted data was discarded in order to reduce the execution time. Computational results presented in [2] showed that good images are obtained although the input data are not sorted. ITATCQ processes the pixels in the same order in which they are read from the original image, that is, from left to right and from top to bottom. Each iteration processes the pixels in the same raster order.

Nevertheless, the results presented in this article show that better quantized images can be obtained when the pixels to be processed are taken randomly from the original image. Therefore, the improved ITATCQ processes the pixels randomly, but ensuring that each iteration processes each pixel exactly once. Computational results show that although this may slightly increase the execution time, it also greatly increases the quality of the quantized image.

5. Computational experiments

The proposed method was coded in C language and executed on a PC running a Linux operating system with 8 GBytes of RAM and AMD Ryzen 5 1600 processor (3.2 GHz). It was applied to twelve color images shown in Fig. 1, which were downloaded from [41,42]. Tests were performed considering four palette sizes: $Q_{max} = \{32, 64, 128, 256\}$, and a linear increase of α from an initial value to 1. The initial values considered were those proposed in [2] to test ITATCQ: $\alpha = \{0.25, 0.30, 0.35, 0.40, 0.45, 0.50\}$. The quality of the quantized image was determined by computing the mean squared error (MSE) by Eq. (3), where p_{ij} is the pixel in the row i, column j from the original image. p'_{ij}, on the other hand, is the pixel in the same position but in the quantized image. In addition, a denotes the image height, whereas b denotes the image width.

$$MSE = \frac{1}{ab}\sum_{i=1}^{a}\sum_{j=1}^{b}\|p_{ij} - p'_{ij}\|^2 \tag{3}$$

To differentiate the references to the original ITATCQ and to the variant proposed in this article, in this section they are denoted as ITATCQ-o and ITATCQ-n, respectively.

Table 2
MSE (*min*: minimum, *av*: average, *dev*: standard deviation) and average time (milliseconds) for ITATCQ-n at iteration 10.

		32 colors				64 colors				128 colors				256 colors			
	α	*min*	*av*	*dev*	*T*	*min*	*av*	*dev*	*T*	*min*	*av*	*dev*	*T*	*min*	*av*	*dev*	*T*
Sailb.	0.25	82.59	102.62	18.7	1261	38.75	41.70	2.2	1829	22.05	22.43	0.6	3049	12.81	13.32	0.4	4990
	0.30	76.19	87.08	6.8	1315	42.43	46.99	3.6	1937	21.58	23.13	1.0	3053	12.86	13.17	0.2	4948
	0.35	73.04	79.12	3.6	1290	39.17	43.19	2.3	1899	23.30	26.25	2.2	3082	13.01	13.32	0.2	5030
	0.40	72.55	78.47	4.2	1276	38.23	40.09	1.4	1899	22.13	23.23	0.8	3096	13.78	14.04	0.3	5167
	0.45	67.09	74.18	3.8	1274	37.09	38.37	1.0	1891	21.54	22.39	0.5	3104	13.16	13.40	0.2	5332
	0.50	69.21	74.49	4.0	1301	36.35	38.08	1.2	1895	21.49	21.85	0.2	3099	12.69	12.92	0.2	5254
			82.66		**1286**		**41.40**		**1892**		**23.21**		**3080**		**13.36**		**5120**
Motor.	0.25	201.49	229.63	27.4	1273	116.48	123.95	6.1	1810	66.51	69.12	1.9	2881	40.70	41.26	0.4	4998
	0.30	209.72	229.29	16.7	1312	136.55	141.51	5.4	1869	67.46	69.30	1.5	2975	40.29	40.91	0.4	5078
	0.35	205.89	227.49	13.3	1314	124.01	131.27	4.3	1888	69.14	73.89	3.7	3084	40.60	41.48	0.5	5131
	0.40	198.27	216.69	14.5	1297	116.39	125.09	6.2	1890	67.44	71.10	2.3	3128	42.03	43.27	0.9	5351
	0.45	194.43	210.31	7.8	1322	111.47	116.65	3.9	1977	66.40	68.44	1.9	3121	41.33	41.95	0.5	5526
	0.50	198.54	206.94	7.3	1327	110.62	117.36	3.8	1903	65.83	67.96	1.3	3130	40.72	41.46	0.8	5421
			220.06		**1307**		**125.97**		**1889**		**69.97**		**3053**		**41.72**		**5251**
Lady	0.25	122.73	137.01	14.8	1299	65.75	70.11	3.2	1834	36.55	38.29	1.1	2831	21.54	22.36	0.5	4885
	0.30	129.52	147.11	15.3	1388	67.12	72.70	3.5	1895	36.41	37.70	0.8	2940	21.21	21.84	0.4	4890
	0.35	122.43	130.96	7.5	1370	67.20	69.97	2.6	1919	38.32	39.90	1.1	3027	20.69	21.43	0.5	4936
	0.40	118.81	130.56	8.0	1359	63.74	68.60	2.9	1921	38.52	39.88	1.2	3087	21.13	21.89	0.4	5015
	0.45	119.84	128.49	6.6	1340	62.99	65.70	1.3	1900	36.71	38.02	1.2	3101	22.35	23.10	0.5	5193
	0.50	117.81	125.35	8.1	1362	63.38	66.22	2.9	1932	35.72	37.07	0.8	3110	21.16	21.91	0.6	5252
			133.25		**1353**		**68.88**		**1900**		**38.48**		**3016**		**22.09**		**5029**
Caps	0.25	179.17	206.40	28.5	1299	86.42	92.97	4.8	1840	42.79	44.65	1.3	2943	23.05	24.54	1.4	5026
	0.30	172.45	190.36	12.5	1345	89.50	98.12	5.6	1971	47.09	49.74	1.8	3047	23.07	24.06	0.7	5064
	0.35	164.55	177.71	10.8	1356	82.46	91.02	7.7	1935	46.41	51.40	3.7	3124	23.62	24.25	0.4	5170
	0.40	169.19	186.50	15.2	1333	78.36	88.96	7.0	1917	44.55	46.49	1.4	3231	25.52	26.84	1.1	5378
	0.45	159.17	179.45	17.4	1335	79.89	83.44	2.4	1927	43.27	44.33	0.9	3163	22.93	24.44	0.7	5373
	0.50	162.85	181.57	15.4	1327	78.14	83.42	2.6	1953	40.71	42.66	1.1	3161	22.27	23.48	0.7	5421
			187.00		**1332**		**89.65**		**1924**		**46.55**		**3111**		**24.60**		**5239**
Parrots	0.25	268.08	289.74	16.5	1290	139.75	148.85	6.7	1858	82.71	85.77	3.5	2868	45.45	47.35	0.9	5016
	0.30	264.30	279.19	17.9	1363	140.71	152.60	9.7	1974	81.78	86.10	6.1	2954	45.89	48.21	1.9	4965
	0.35	245.37	262.82	12.8	1334	144.50	150.95	5.2	1990	80.13	86.40	4.4	3048	46.82	47.71	0.7	5103
	0.40	240.16	258.22	15.6	1332	137.04	150.52	11.8	1977	82.55	87.67	2.7	3094	48.41	50.59	1.7	5212
	0.45	240.26	252.62	10.5	1305	135.53	146.95	8.4	1961	80.76	85.84	4.4	3052	47.21	48.85	1.1	5370
	0.50	240.29	258.46	15.5	1320	135.50	146.07	8.1	1939	78.03	83.61	3.9	3081	45.15	48.66	2.9	5533
			266.84		**1324**		**149.32**		**1950**		**85.90**		**3016**		**48.56**		**5200**
Girl	0.25	137.37	179.18	55.5	1258	79.75	84.98	3.9	1784	43.07	44.64	1.1	2827	25.41	26.43	0.7	5143
	0.30	132.32	147.21	14.7	1310	76.85	87.32	11.5	1861	43.44	45.61	1.5	2859	25.29	26.29	0.7	5041
	0.35	134.63	146.60	9.6	1308	80.92	86.31	4.5	1917	45.79	48.93	2.7	3019	26.55	26.92	0.4	5089
	0.40	131.69	145.79	11.8	1326	78.64	81.06	2.7	1930	45.46	46.45	1.1	3052	27.10	27.96	0.5	5735
	0.45	131.98	141.69	7.6	1337	75.50	79.09	2.7	1918	43.02	44.70	1.1	3083	26.05	26.93	0.7	5617
	0.50	131.25	140.83	8.9	1352	73.74	80.45	6.0	1924	43.39	44.52	1.2	3110	25.31	26.00	0.4	5465
			150.22		**1315**		**83.20**		**1889**		**45.81**		**2992**		**26.76**		**5348**

The initial experiments analyzed the effect of the number of iterations used to update α. For this purpose, IT was set to 25 and ITATCQ-n was applied, updating α during several of the initial iterations: first 2 iterations, first 4 iterations, first 6 iterations, first 10 iterations and all the iterations. Twenty independent tests were executed for each problem, palette size, initial α value and number of iterations for which this parameter increases. Figs. 2 and 3 show the average MSE obtained for the six initial α values considered; the results of the initial iterations are not represented because several large values are obtained that make the information shown by the subfigures less clear. It is observed that better images are obtained when α is updated during the 2 or 4 initial iterations. Certainly, as described in [2], ITATCQ-o obtains the largest reduction in the MSE error for the second iteration. Since the case that updates α during the 2 initial iterations generates the worst result for several images and palette sizes, the case that applies the update during 4 iterations was selected and its results are used in the following discussion. Tables 1 and 2 include the results of this case at iteration 10. This iteration was selected because it generates good results in comparison with other color quantization methods discussed later in this section. The tables show in bold the average MSE and the average execution time for each image and palette size. In addition, the best quantized image obtained at iteration 10 for each palette size is included as supplementary material.

The results presented in Tables 1 and 2 show that a quantized image with up to 256 colors can be obtained in less than 5.8 s when images with 393,216 pixels are analyzed, and this value reduces to 3.6 s when the images include 262,144 pixels. In general, better quantized images are obtained when the initial α considered is large, although the images obtained for any α value may be acceptable (the error is not too large for any α value). It is also observed that as the palette size increases, the differences in the errors obtained for each of the six initial α values decrease.

Tables 3 and 4 show the results of ITATCQ-o for the same α values and iterations considered in Tables 1 and 2 to test ITATCQ-n. These tables include the number of colors of the quantized palette, because Q_{max} is not always reached when α is small. It can be observed that ITATCQ-n generates better quantized images than ITATCQ-o for all the images and palette sizes. On the other hand, although the original method generates bad images for some α values, the new variant obtains better images for these cases; this can especially be observed for cases where $\alpha = 0.25$ and $\alpha = 0.50$. When α is small, ITATCQ-o generates images with less than Q_{max} colors; therefore, the error associated with the quantized image is large. However, this problem disappears when ITATCQ-n is applied. Regarding the execution time, ITATCQ-n consumes a little more time than ITATCQ-o due to two

Table 3
MSE and average time (milliseconds) for ITATCQ-o at iteration 10 (q: number of colors of the quantized palette; Q_{max}: limit to the number of colors of the quantized palette).

		$Q_{max}=32$			$Q_{max}=64$			$Q_{max}=128$			$Q_{max}=256$		
	α	q	MSE	T	q	MSE	T	q	MSE	T	q	MSE	T
Lenna	0.25	32	141.68	553	43	126.26	635	43	126.26	629	43	126.26	623
	0.30	32	140.30	573	64	80.58	940	119	61.20	1331	119	61.20	1313
	0.35	32	148.11	559	64	83.91	965	128	51.03	1687	256	37.50	2532
	0.40	32	158.71	562	64	95.08	958	128	50.96	1720	256	32.58	3119
	0.45	32	186.61	567	64	95.92	962	128	55.11	1721	256	32.72	3169
	0.50	32	198.61	556	64	116.54	989	128	64.03	1721	256	35.83	3199
			162.34	**562**		**99.72**	**908**		**68.10**	**1468**		**54.35**	**2326**
Peppers	0.25	32	320.19	549	58	215.31	830	58	215.31	836	58	215.31	811
	0.30	32	337.68	559	64	191.64	940	108	151.23	1217	108	151.23	1190
	0.35	32	290.21	558	64	158.93	965	128	98.15	1741	239	75.05	2566
	0.40	32	312.88	547	64	167.05	975	128	96.08	1807	256	64.63	3110
	0.45	32	355.85	535	64	168.93	990	128	100.82	1786	256	61.69	3160
	0.50	32	523.19	534	64	205.07	952	128	114.04	1792	256	64.39	3247
			356.67	**547**		**184.49**	**942**		**129.27**	**1530**		**105.38**	**2347**
Mandrill	0.25	1	9168.97	150	1	9168.97	150	1	9168.97	151	1	9168.97	151
	0.30	32	404.44	570	64	265.53	886	82	242.48	954	82	242.48	960
	0.35	32	457.09	577	64	276.38	955	128	171.61	1640	256	121.95	2501
	0.40	32	444.64	584	64	313.44	977	128	172.85	1747	256	114.54	3096
	0.45	32	440.91	595	64	317.94	978	128	190.55	1816	256	112.59	3235
	0.50	32	429.65	580	64	303.70	971	128	207.16	1749	256	121.35	3174
			1890.95	**509**		**1774.32**	**820**		**1692.27**	**1343**		**1646.98**	**2186**
Plane	0.25	32	110.21	542	51	75.69	754	51	75.69	698	51	75.69	702
	0.30	32	99.04	596	64	55.43	916	78	52.90	973	78	52.90	1012
	0.35	32	109.46	561	64	46.43	1003	128	31.01	1499	145	30.20	1576
	0.40	32	104.47	564	64	48.07	934	128	26.96	1667	256	18.59	2785
	0.45	32	135.17	630	64	80.48	1000	128	31.33	1696	256	17.33	3171
	0.50	32	143.78	557	64	84.33	975	128	54.83	1760	256	18.30	3251
			117.02	**575**		**65.07**	**930**		**45.45**	**1382**		**35.50**	**2083**
Blond	0.25	32	103.08	553	64	80.90	827	65	80.77	812	65	80.77	812
	0.30	32	122.85	569	64	64.39	960	105	51.41	1314	105	51.41	1312
	0.35	32	102.86	571	64	66.42	946	128	37.74	1694	189	32.86	2062
	0.40	32	133.56	572	64	63.07	962	128	38.84	1724	256	25.90	3031
	0.45	32	158.01	573	64	80.22	959	128	38.91	1713	256	21.37	3223
	0.50	32	185.26	558	64	107.20	958	128	38.61	1729	256	22.11	3205
			134.27	**566**		**77.03**	**935**		**47.71**	**1498**		**39.07**	**2274**
Lake	0.25	32	244.16	562	59	170.10	858	59	170.10	861	59	170.10	854
	0.30	32	225.84	578	64	154.54	969	128	98.77	1582	133	98.09	1579
	0.35	32	257.74	553	64	162.31	966	128	101.60	1806	256	65.08	2811
	0.40	32	300.25	551	64	169.79	969	128	104.01	1753	256	63.25	3236
	0.45	32	334.10	550	64	173.75	953	128	103.46	1751	256	64.69	3176
	0.50	32	737.98	551	64	189.91	962	128	112.91	1724	256	66.31	3178
			350.01	**558**		**170.07**	**946**		**115.14**	**1580**		**87.92**	**2472**

factors: the random selection of the ants and the number of colors of the quantized palette. ITATCQ-n obtains quantized palettes with Q_{max} colors during the initial iterations, and this makes each iteration consume more time than when the palette includes fewer colors. Tables 3 and 4 clearly show that ITATCQ-o consumes less time for the cases that define quantized palettes with less than Q_{max} colors.

To complete the comparison of both ITATCQ variants, Fig. 4 shows the results obtained by ITATCQ-n and ITATCQ-o as the iterations proceed. To reduce the length of the article, this figure only shows the case corresponding to 256 colors. However, the figures corresponding to 32, 64 and 128 colors are included as supplementary material. These figures show the average MSE obtained for all the initial α values considered, so as to make the results easier to interpret. It is observed that, although ITATCQ-n obtains large MSE errors at iteration 1 for some images, this value decreases rapidly during the initial iterations (2 or 3 iterations). In contrast, ITATCQ-o usually obtains smaller errors than ITATCQ-n at iteration 1, but the error is worse than that obtained by ITATCQ-n as the iterations proceed.

The following two subsections analyze the effect of each element included in the new variant of ITATCQ, in order to demonstrate that better results are obtained when the increase of α is combined with the random selection of the pixels.

5.1. The effect of the α parameter

Table 5 shows the results obtained when ITATCQ is applied with variable α but the pixels are processed in the order in which they are in the original image.

It is observed that this case generates better global images than ITATCQ-o. It should be noted that the results are the same for both variants from a certain α value for which the palette includes Q_{max} colors. This occurs when Q_{max} colors are included in the palette during the first iteration. However, after 10 iterations of the method reported in Table 5, the quantized palette always includes Q_{max} colors and this generates better quantized palettes for the cases that consider smaller initial α values, while ITATCQ-o generates poor quality images for said cases.

When Table 5 is compared to Tables 1 and 2, the average MSE obtained for each problem and palette size is always better when ITATCQ-n is considered. Moreover, when the result for each α value is compared, the variant that does not consider random pixels only obtains better MSE for 20 out of 288 cases (12 images × 4 palettes × 6 α values =288).

Table 4
MSE and average time (milliseconds) for ITATCQ-o at iteration 10 (q: number of colors of the quantized palette; Q_{max}: limit to the number of colors of the quantized palette).

		$Q_{max}=32$			$Q_{max}=64$			$Q_{max}=128$			$Q_{max}=256$		
	α	q	MSE	T	q	MSE	T	q	MSE	T	q	MSE	T
Sailb.	0.25	32	74.99	828	64	44.41	1382	113	33.80	2028	113	33.80	1998
	0.30	32	80.34	833	64	45.00	1387	128	26.36	2501	211	18.50	3282
	0.35	32	126.22	818	64	42.40	1422	128	25.13	2538	256	14.30	4507
	0.40	32	112.70	835	64	49.34	1416	128	24.54	2551	256	14.50	4604
	0.45	32	138.98	825	64	85.81	1391	128	27.84	2574	256	15.39	4632
	0.50	32	176.62	830	64	84.25	1386	128	35.39	2592	256	16.20	4659
			118.31	**828**		**58.54**	**1397**		**28.84**	**2464**		**18.78**	**3947**
Motor.	0.25	32	243.62	839	64	165.88	1464	100	127.61	1954	100	127.61	1946
	0.30	32	226.82	849	64	128.60	1442	128	79.39	2634	215	58.45	3835
	0.35	32	238.37	829	64	135.69	1441	128	87.14	2570	256	50.66	4891
	0.40	32	271.09	833	64	153.54	1424	128	79.11	2586	256	49.08	4748
	0.45	32	322.32	835	64	165.63	1436	128	86.25	3028	256	47.09	4759
	0.50	32	322.70	859	64	168.61	1422	128	91.77	2623	256	50.08	4715
			270.82	**841**		**152.99**	**1438**		**91.88**	**2566**		**63.83**	**4149**
Lady	0.25	32	137.52	821	64	73.28	1413	98	59.28	1821	98	59.28	1827
	0.30	32	145.97	834	64	74.44	1420	128	44.18	2496	149	42.22	2681
	0.35	32	182.48	828	64	82.43	1491	128	43.54	2623	242	29.08	3910
	0.40	32	192.58	813	64	94.98	1474	128	43.62	2574	256	24.34	4688
	0.45	32	202.83	821	64	124.09	1414	128	56.95	2583	256	23.61	4807
	0.50	32	181.63	850	64	154.53	1446	128	72.95	2561	256	30.57	4717
			173.83	**828**		**100.62**	**1443**		**53.42**	**2443**		**34.85**	**3772**
Caps	0.25	32	227.40	813	64	98.75	1381	121	60.60	2170	121	60.60	2169
	0.30	32	265.31	821	64	105.35	1401	128	51.92	2487	235	33.46	3763
	0.35	32	259.83	807	64	151.87	1416	128	53.49	2489	256	27.36	4594
	0.40	32	269.66	818	64	128.38	1372	128	70.89	2486	256	28.47	4617
	0.45	32	352.93	820	64	166.80	1385	128	86.96	2486	256	35.34	4686
	0.50	32	466.67	864	64	214.19	1388	128	145.86	2513	256	56.86	4740
			306.97	**824**		**144.22**	**1391**		**78.29**	**2439**		**40.35**	**4095**
Parrots	0.25	32	270.66	816	64	157.72	1347	69	152.17	1412	69	152.17	1431
	0.30	32	292.16	809	64	176.46	1370	128	92.98	2469	147	89.92	2609
	0.35	32	335.47	888	64	201.25	1419	128	106.27	2542	256	52.83	4627
	0.40	32	297.61	823	64	204.56	1404	128	125.05	2568	256	62.30	4822
	0.45	32	297.53	855	64	208.99	1427	128	135.79	2535	256	83.55	4917
	0.50	32	292.95	871	64	219.24	1411	128	156.75	2929	256	86.90	4804
			297.73	**844**		**194.70**	**1396**		**128.17**	**2409**		**87.94**	**3868**
Girl	0.25	32	226.73	793	64	113.49	1392	126	62.67	2349	126	62.67	2302
	0.30	32	211.49	799	64	120.46	1424	128	53.29	2571	228	36.52	3753
	0.35	32	245.94	909	64	105.58	1381	128	74.74	2548	256	29.34	4642
	0.40	32	303.85	805	64	125.07	1395	128	81.36	2557	256	35.26	4767
	0.45	32	338.52	794	64	139.83	1377	128	76.16	2530	256	44.22	4680
	0.50	32	356.55	845	64	133.07	1382	128	77.07	3082	256	47.56	4666
			280.51	**824**		**122.92**	**1392**		**70.88**	**2606**		**42.59**	**4135**

5.2. The effect of the random selection of the pixels

Tables 6 and 7 show the results obtained when ITATCQ is applied with constant α, but the pixels are processed randomly. It is clearly observed that the results obtained in this case are very bad when α is small, because palettes with very few colors are defined in many cases. It is also observed that the MSE improves as α increases, since palettes with more colors are obtained.

When the results of Tables 6 and 7 are compared to those of ITATCQ-o, the MSE values are better for the cases that consider larger α values, but worse in the others. Although in some cases ITATCQ-o also generates palettes with less than Q_{max} colors, it includes more colors than the variant analyzed in this subsection.

When the results of Tables 6 and 7 are compared to those of ITATCQ-n, although the quality of the images is better in some specific cases, it is much worse and includes fewer colors for several α values. In addition, the average MSE obtained for each problem and palette size is clearly better when the results of ITATCQ-n are considered. When the execution time of Tables 1–2 and Tables 6–7 is compared, similar values are obtained for the cases with Q_{max} colors, which confirms that the execution time of ITATCQ-n increases slightly with respect to ITATCQ-o due to the random selection of the pixels that are going to be processed.

In sum, the case than only considers random pixels generates worse images than the case that considers both random pixels and variable α value.

5.3. The function used to update α

The results presented in this section consider a linear increase of α over the course of the iterations, so that each time α is updated the same increment is applied. However, other possibilities were analyzed before selecting the linear function:

- a greater increase for the initial iterations and a smaller increase for the final iterations.
- a smaller increase for the initial iterations and larger increase for the final iterations.

The reduction in the error obtained by ITATCQ-o during the iterative process decreases as the iterations progress [2]. For this reason, it seems interesting to center the effect of the increase of α on the initial iterations. Fig. 5 shows the shape of two functions corresponding to the two cases proposed above, labeled M1 and M2, respectively. Both functions were selected taking into account the initial values of α used in the tests, and that this parameter must not exceed 1.

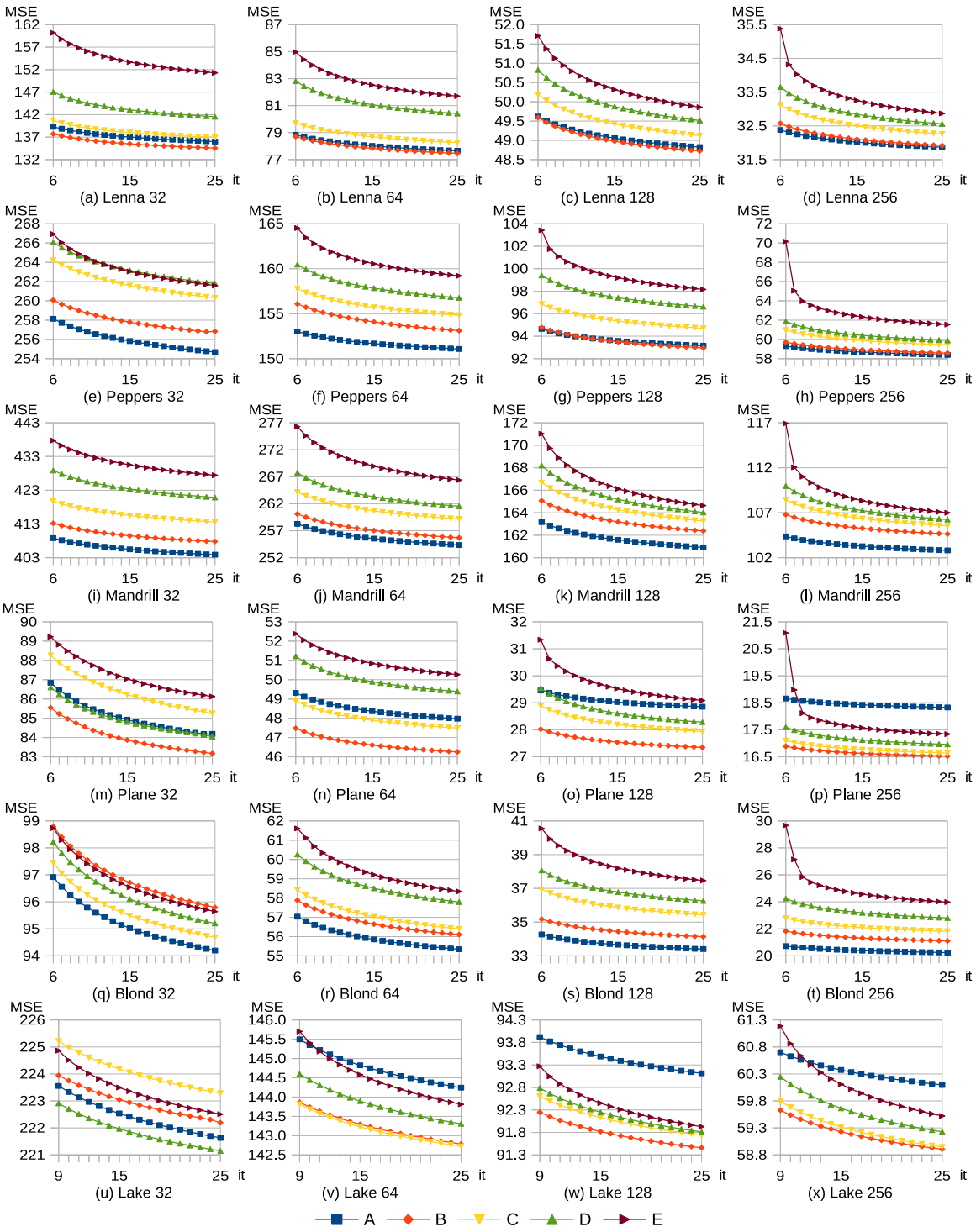


Fig. 2. Average MSE for ITATCQ-n with the increase of α applied during several initial iterations (A: 2 iterations; B: 4 iterations; C: 6 iterations; D: 10 iterations; E: 25 iterations).

Figs. 6 and 7 compare the results obtained with the three functions used to update α. The results shown for the linear case (labeled M3 in the figure) are those corresponding to the initial tests described in this section that apply a linear increase during the 4 initial iterations. These figures clearly show that better global results are obtained when a linear increase is applied. It is also observed that the worst results are generated when the increments are small during the initial iterations (case M2).

5.4. Comparison with other color quantization methods

The results of ITATCQ-n were compared to those of the following methods: Binary splitting (BS), the method proposed by Wu in [9] (WU), Octree (OC), Variance-based method (VB), Median-cut (MC), Neuquant (NQ), K-means (KM), ATCQ, ATCQ+FA, BS+ATCQ, ABC+ATCQ and SFLA. Tables 8 to 9 show the results of 20 independent tests for each image and palette size.

Table 8 shows the results of the iterative methods, for which 10 iterations were performed. On the other hand, for ATCQ and the methods based on it (BS+ATCQ, ABC+ATCQ and ATCQ+FA), the variant of

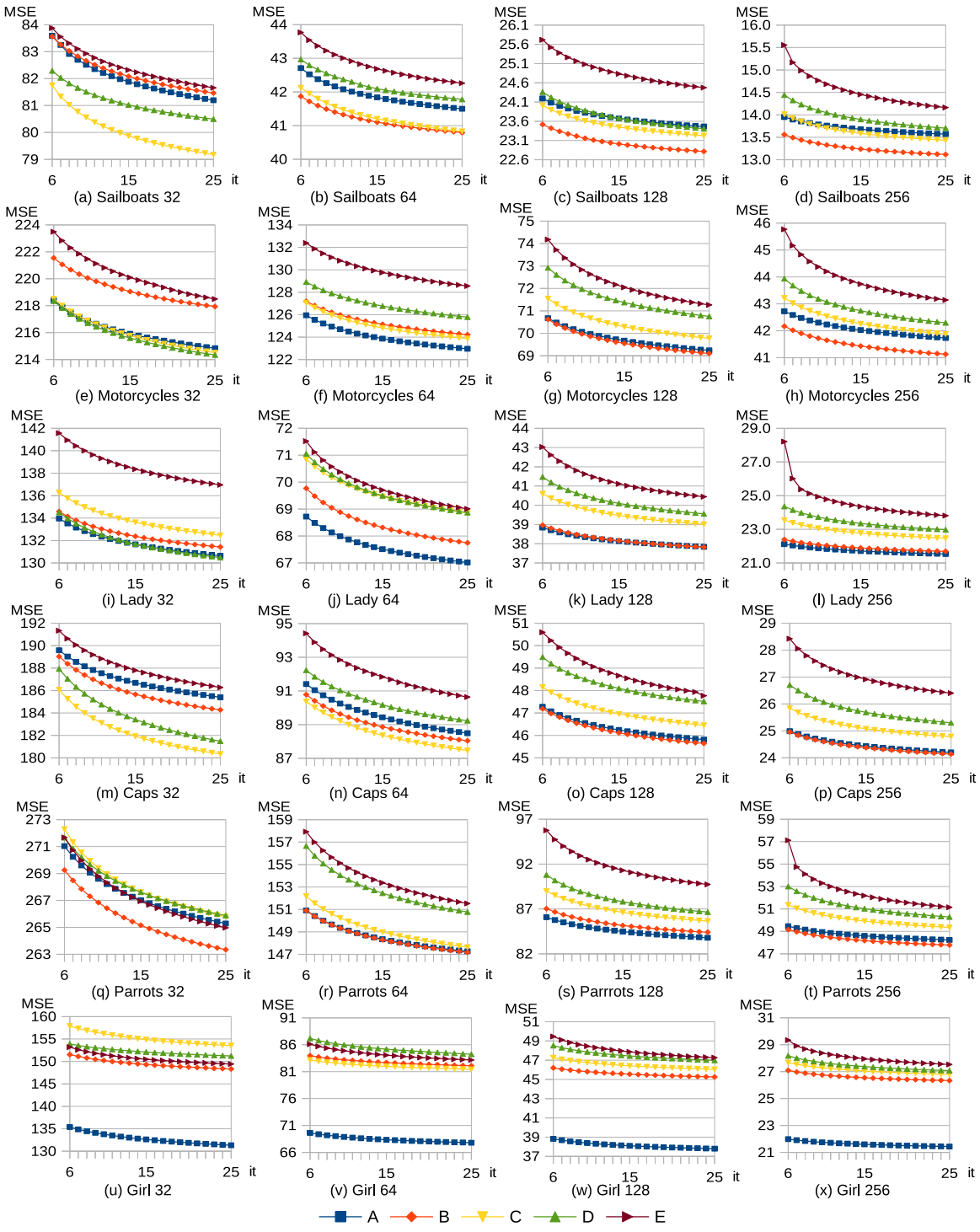


Fig. 3. Average MSE for ITATCQ-n with the increase of α applied during several initial iterations (A: 2 iterations; B: 4 iterations; C: 6 iterations; D: 10 iterations; E: 25 iterations).

this algorithm that builds a 3-level tree was considered and the initial α values selected to test ITATCQ-n were used. Given that 6 α values were selected, this requires considering 6 food sources for ABC+ATCQ and 6 fireflies for ATCQ+FA. With respect to SFLA, it was tested considering 6 frogs, 2 memeplexes and a step for sampling the input image equal to 10, which generated good results in [34].

It can be observed that the average MSE results of ITATCQ-n shown in bold in Table 1 are always better than those generated by WU, OC, VB, MC, NQ and ATCQ, and are also better than those of BS and ATCQ+FA for almost all cases (only three cases are not improved when ATCQ+FA is considered and only two when BS is considered). Additionally, KM is improved in 38 out of 48 cases, and the percentage of error reduction obtained by KM for the 10 unimproved cases ranges between 0.2% and 4.1%. It is also observed that ITATCQ-n consumes less time than KM and ATCQ+FA, but more than the other methods compared in this paragraph. This difference in runtime is common when comparing splitting methods and clustering-based methods. In any case, ITATQ-n can obtain better results than these faster methods even considering less than 10 iterations (therefore, consuming less time). For example, when the third iteration of ITATCQ-n is considered, the average error of the images generated by this method is always better than the result obtained by OC, VB, MC and ATCQ. In addition,

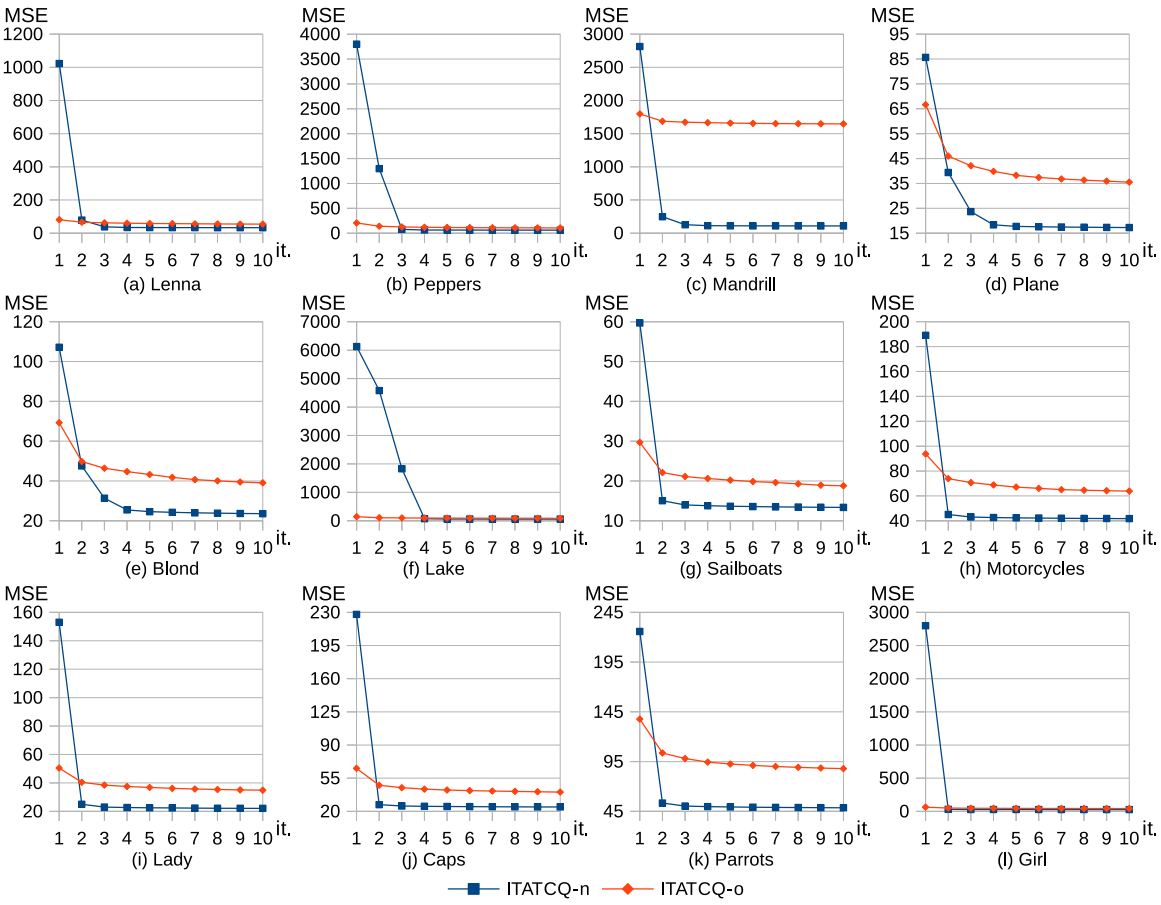


Fig. 4. Evolution of the average MSE of ITATCQ-n and ITATCQ-o during the 10 initial iterations — 256 colors.

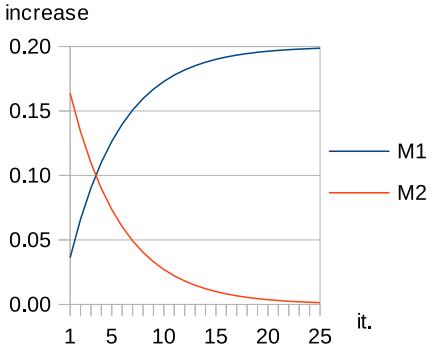


Fig. 5. Two functions used to define the increment of α: M1: $0.2e^{-0.2it}$, M2: $0.2-0.2e^{-0.2it}$ (it: iteration of the algorithm).

it is better than the result of WU, NQ and BS for 47, 46 and 43 out of 48 cases, respectively. The results of the third iteration of ITATCQ-n are included as supplementary material.

SFLA and ABC+ATCQ generate better images than ITATCQ-n in some cases, but consume much more time (especially the second method). When the best average MSE obtained by ITATCQ-n for each problem and palette size is considered, SFLA obtains better images for 22 cases, and the percentage of error reduction ranges in between 0.3% and 8.2%, although it exceeds 5% only in 3 cases. On the other hand, ABC+ATCQ generates better images for 42 cases, with a reduction in the error ranging in the interval $[0.8\%, 13.3\%]$, but 17 of the cases do not reach 5.8%.

BS+ATCQ is the method that generates the best images among all the methods reported in Tables 8 to 9. It is observed that the best average MSE obtained by ITATCQ-n for each problem and palette size only improves the results of BS+ATCQ for 8 cases. Nevertheless, the reduction in the error obtained by BS+ATCQ is less than 1% for 12 cases and, although it reaches 11.3% as a maximum, only 9 cases exceed 5%. BS+ATCQ can generate a good image faster than ITATCQ-n, but ITATCQ-n can obtain better images than BS+ATCQ when more iterations are performed. Certainly, if the results of ITATCQ-n at iteration 25 are considered, the proposed method improves the image generated by BS+ATCQ in 17 cases and the difference is smaller than 3% for other 18 cases. The results of this iteration of ITATCQ-n are included as supplementary material.

6. Conclusion

The ITATCQ algorithm is a color quantization method that represents the pixels of the image by ants that build a tree. This is an iterative method that considers a parameter, α, which is used to determine when a new color of the palette must be defined and when an ant must be associated with an element already existing in the palette. The value of said parameter influences the quality of the final image and must be selected according to the size of the quantized palette.

This article has analyzed the effect of two modifications on the original algorithm: the random processing of the pixels of the original image and the increase of the value of α along the iterations. Computational results show that better images are always obtained when the method considers both modifications. Moreover, the initial value selected for the α parameter reduces its influence on the final image.

This new variant of ITATCQ has been compared to other color quantization methods, generating good results. Regarding the execution time, the proposed method is faster than the other iterative methods analyzed. In addition, it generates better images than some of these

Table 5
MSE and average time (milliseconds) for ITATCQ with α linearly increased — iteration 10.

		32 colors		64 colors		128 colors		256 colors				32 colors		64 colors		128 colors		256 colors	
	α	MSE	T	MSE	T	MSE	T	MSE	T		α	MSE	T	MSE	T	MSE	T	MSE	T
Lenna	0.25	131.84	559	90.87	950	60.43	1674	34.95	2985	Plane	0.25	110.21	544	69.66	885	37.04	1624	18.06	2868
	0.30	140.30	576	79.67	966	58.19	1650	36.97	3122		0.30	99.04	560	54.02	945	35.52	1637	21.27	2907
	0.35	148.11	566	83.91	994	51.04	1716	35.72	3080		0.35	109.46	553	46.43	973	32.58	1654	22.18	2961
	0.40	158.71	571	95.08	985	50.96	1738	32.56	3234		0.40	104.47	546	48.07	947	26.96	1663	21.03	3050
	0.45	186.61	567	95.92	990	55.11	1788	32.72	3243		0.45	135.17	561	80.48	967	31.33	1704	17.42	3120
	0.50	198.61	564	116.54	979	64.03	1763	35.83	3250		0.50	143.78	549	84.33	950	54.83	1723	18.30	3160
		160.70	**567**	**93.66**	**977**	**56.63**	**1722**	**34.79**	**3152**			**117.02**	**552**	**63.83**	**945**	**36.38**	**1668**	**19.71**	**3011**
Peppers	0.25	320.19	557	209.20	923	104.79	1624	63.84	2997	Blond	0.25	103.08	563	83.83	935	43.07	1617	23.72	2999
	0.30	337.68	568	234.10	945	107.71	1643	63.92	2994		0.30	122.85	581	64.39	971	40.32	1617	24.36	3025
	0.35	290.21	561	158.93	963	98.15	1740	63.30	3123		0.35	102.86	582	66.42	966	37.51	1706	24.14	3099
	0.40	312.88	564	167.05	958	96.08	1793	64.53	3154		0.40	133.56	580	63.07	970	38.84	1726	24.08	3151
	0.45	355.85	549	168.93	959	100.82	1731	61.69	3261		0.45	158.01	594	80.22	975	38.91	1713	21.37	3294
	0.50	523.19	537	205.07	953	114.04	1798	64.39	3221		0.50	185.26	585	107.20	969	38.61	1733	22.11	3254
		356.67	**556**	**190.55**	**950**	**103.60**	**1722**	**63.61**	**3125**			**134.27**	**581**	**77.52**	**964**	**39.54**	**1685**	**23.30**	**3137**
Mandrill	0.25	457.40	550	302.78	924	200.82	1593	121.36	2968	Lake	0.25	244.16	569	182.98	950	107.86	1649	63.35	3007
	0.30	404.44	583	259.74	958	175.84	1640	117.89	3070		0.30	225.84	585	154.54	962	107.13	1705	66.74	3018
	0.35	457.09	588	276.38	976	172.64	1646	126.81	3083		0.35	257.74	566	162.31	966	101.60	1743	68.62	3167
	0.40	444.64	597	313.44	991	172.85	1747	108.13	3254		0.40	300.25	564	169.79	988	104.01	1739	66.03	3246
	0.45	440.91	603	317.94	1003	190.55	1756	112.59	3278		0.45	334.10	587	173.75	965	103.46	1751	64.69	3230
	0.50	429.65	590	303.70	986	207.16	1748	121.35	3317		0.50	737.98	570	189.91	966	112.91	1726	66.31	3226
		439.02	**585**	**295.66**	**973**	**186.64**	**1688**	**118.02**	**3162**			**350.01**	**574**	**172.21**	**966**	**106.16**	**1719**	**65.95**	**3149**
Sailb.	0.25	74.99	834	44.41	1400	32.84	2432	16.83	4557	Caps	0.25	227.40	797	98.75	1391	58.03	2664	33.76	4588
	0.30	80.34	828	45.00	1393	26.36	2500	15.89	4603		0.30	265.31	812	105.35	1391	51.92	2571	36.53	4720
	0.35	126.22	831	42.40	1432	25.13	2549	13.62	4576		0.35	259.83	795	151.87	1422	53.49	2874	27.36	4703
	0.40	112.70	833	49.34	1416	24.54	2558	14.50	4648		0.40	269.66	794	128.38	1384	70.89	2540	28.47	5273
	0.45	138.98	826	85.81	1415	27.84	2563	15.39	5005		0.45	352.93	802	166.80	1384	86.96	2521	35.34	5127
	0.50	176.62	828	84.25	1398	35.39	2805	16.20	4706		0.50	466.67	798	214.19	1418	145.86	2542	56.86	4772
		118.31	**830**	**58.54**	**1409**	**28.68**	**2568**	**15.40**	**4683**			**306.97**	**800**	**144.22**	**1398**	**77.86**	**2619**	**36.39**	**4864**
Motor.	0.25	243.62	848	165.88	1462	94.14	2486	48.07	4514	Parrots	0.25	270.66	799	158.57	1349	106.70	2419	71.40	4413
	0.30	226.82	856	128.60	1451	79.39	2610	45.99	4573		0.30	292.16	802	176.46	1408	86.06	2502	61.27	4643
	0.35	238.37	848	135.69	1466	87.14	2610	50.66	5302		0.35	335.47	823	201.25	1412	106.27	2526	52.46	4660
	0.40	271.09	875	153.54	1426	79.11	2577	49.08	4992		0.40	297.61	812	204.56	1435	125.05	2562	62.30	4844
	0.45	322.32	907	165.63	1434	86.25	3052	47.09	4842		0.45	297.53	826	208.99	1437	135.79	3129	83.55	4929
	0.50	322.70	922	168.61	1431	91.77	2866	50.08	4767		0.50	292.95	819	219.24	1415	156.75	2698	86.90	4888
		270.82	**876**	**152.99**	**1445**	**86.30**	**2700**	**48.50**	**4832**			**297.73**	**814**	**194.84**	**1409**	**119.44**	**2639**	**69.65**	**4730**
Lady	0.25	137.52	832	73.28	1432	56.33	2541	30.53	4443	Girl	0.25	226.73	800	113.49	1400	57.50	2505	34.16	4650
	0.30	145.97	837	74.44	1433	47.10	2577	29.70	4515		0.30	211.49	799	120.46	1437	53.29	2754	31.81	4756
	0.35	182.48	837	82.43	1432	43.54	2584	29.25	4655		0.35	245.94	790	105.58	1398	74.74	2672	29.34	4706
	0.40	192.58	821	94.98	1444	43.62	2547	24.34	4965		0.40	303.85	803	125.07	1419	81.36	2585	35.26	5247
	0.45	202.83	824	124.09	1424	56.95	2589	23.61	4939		0.45	338.52	798	139.83	1383	76.16	2543	44.22	4739
	0.50	181.63	822	154.53	1500	72.95	2939	30.57	4781		0.50	356.55	805	133.07	1397	77.07	2602	47.56	4830
		173.83	**829**	**100.62**	**1444**	**53.41**	**2630**	**28.00**	**4716**			**280.51**	**799**	**122.92**	**1406**	**70.02**	**2610**	**37.06**	**4821**

methods such as ATCQ+FA and KM. Although the proposed method consumes more time that the non-iterative methods compared, it also obtains better images than most of those methods.

Funding

This work was supported by the Samuel Solórzano Barruso Memorial Foundation of the University of Salamanca, Spain [grant number FS/102015].

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.jvcir.2021.103180.

References

[1] M. Garey, D. Johnson, H. Witsenhausen, The complexity of the generalized Lloyd-max problem (corresp.), IEEE Trans. Inform. Theory 28 (2) (1982) 255–256, http://dx.doi.org/10.1109/TIT.1982.1056488.

[2] M.-L. Pérez-Delgado, An iterative method to improve the results of Ant-tree algorithm applied to colour quantisation, Int. J. Bio-Inspir. Comput. 12 (2) (2018) 87–114, http://dx.doi.org/10.1504/IJBIC.2018.10015516.

[3] M.-L. Pérez-Delgado, Colour quantization with Ant-tree, Appl. Soft Comput. 36 (2015) 656–669, http://dx.doi.org/10.1016/j.asoc.2015.07.048.

[4] P. Heckbert, Color image quantization for frame buffer display, in: Proceedings of the 9th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '82, ACM, New York, NY, USA, 1982, pp. 297–307, http://dx.doi.org/10.1145/800064.801294.

[5] S. Wan, P. Prusinkiewicz, S. Wong, Variance-based color image quantization for frame buffer display, Color Res. Appl. 15 (1) (1990) 52–58, http://dx.doi.org/10.1002/col.5080150109.

[6] M.T. Orchard, C.A. Bouman, Color quantization of images, IEEE Trans. Signal Process. 39 (12) (1991) 2677–2690, http://dx.doi.org/10.1109/78.107417.

[7] M. Gervautz, W. Purgathofer, A simple method for color quantization: Octree quantization, in: A.S. Glassner (Ed.), Graphics Gems, Academic Press Professional, Inc., San Diego, CA, USA, 1990, pp. 287–293.

[8] X. Wu, Efficient statistical computations for optimal color quantization, in: J. Arvo (Ed.), Graphics Gems II, Academic Press, 1991, pp. 126–133.

[9] X. Wu, Color quantization by dynamic programming and principal analysis, ACM Trans. Graph. 11 (4) (1992) 348–372.

Table 6
MSE (*min*: minimum, *av*: average, *dev*: standard deviation) and average time (milliseconds) for ITATCQ that uses constant α and random pixels — iteration 10 (q: number of colors of the quantized palette; Q_{max}: limit to the number of colors of the quantized palette).

		$Q_{max}=32$					$Q_{max}=64$					$Q_{max}=128$					$Q_{max}=256$				
	α	q	*min*	*av*	*dev*	T	q	*min*	*av*	*dev*	T	q	*min*	*av*	*dev*	T	q	*min*	*av*	*dev*	T
Len.	0.25	2	566.97	5374.14	2203.3	371	2	694.31	5331.45	2292.3	401	1	6361.76	6361.76	0.0	366	2	694.04	5331.27	2292.7	411
	0.30	32	147.30	231.20	50.7	777	64	85.87	123.68	37.2	1121	118	56.29	73.35	29.5	1327	125	53.94	72.60	36.9	1386
	0.35	32	129.85	141.88	11.7	803	64	77.51	81.62	2.5	1228	128	51.64	51.99	0.4	1915	256	36.39	37.22	0.6	2908
	0.40	32	121.43	131.36	6.1	815	64	76.63	78.13	1.3	1244	128	49.49	50.49	0.7	2040	256	33.33	33.72	0.29	3475
	0.45	32	123.35	131.18	7.6	859	64	75.36	75.87	0.4	1291	128	48.04	48.84	0.5	2105	256	32.39	32.67	0.2	3541
	0.50	32	118.93	124.62	4.6	853	64	74.45	75.79	1.0	1269	128	47.60	48.47	0.6	2050	256	31.69	32.06	0.2	3525
				1022.40		**746**			**961.09**		**1092**			**1105.81**		**1634**			**923.26**		**2541**
Pep.	0.25	1	9633.84	9633.84	0.0	365	6	244.71	8780.28	2830.9	429	5	237.65	8779.64	2833.1	493	1	9633.84	9633.84	0.0	417
	0.30	7	294.97	7936.47	3776.4	441	7	175.94	8774.03	2851.7	458	10	156.97	8772.30	2857.4	517	29	151.35	7048.25	4428.3	738
	0.35	7	266.55	7931.58	3787.3	433	7	173.73	8773.83	2852.3	449	47	107.98	6170.50	4805.2	1050	57	83.49	7030.89	4458.1	994
	0.40	32	247.27	267.10	14.3	830	64	143.61	158.62	7.9	1293	128	94.90	101.50	4.4	2296	256	62.10	64.73	1.6	3618
	0.45	32	251.51	265.10	15.4	864	64	147.38	155.43	5.1	1338	128	91.55	95.19	2.9	2351	256	59.28	60.29	1.0	3797
	0.50	32	240.31	253.10	9.2	871	64	145.78	151.65	4.6	1329	128	88.58	92.52	2.4	2346	256	57.44	58.57	0.9	3799
				4381.20		**634**			**4465.64**		**883**			**4001.94**		**1509**			**3982.76**		**2227**
Man.	0.25	1	9168.97	9168.97	0.0	365	1	9168.97	9168.97	0.0	380	2	930.86	8420.05	2483.9	414	1	9168.97	9168.97	0.0	369
	0.30	15	457.01	4845.55	4340.3	570	1	9168.97	9168.97	0.0	373	1	9168.97	9168.97	0.0	390	32	260.01	5935.64	4486.1	663
	0.35	32	402.27	440.86	28.8	826	64	258.75	274.89	12.8	1224	128	167.41	174.05	4.3	1864	254	117.84	122.40	3.7	2666
	0.40	32	397.85	411.41	19.0	826	64	250.06	262.63	8.7	1246	128	162.06	164.81	2.5	2054	256	108.49	110.41	0.8	3454
	0.45	32	392.58	405.07	11.9	833	64	245.23	255.46	7.8	1311	128	159.85	164.00	3.2	2078	256	105.91	107.10	1.0	3595
	0.50	32	385.81	399.32	13.0	838	64	243.33	252.39	5.9	1297	128	159.31	162.13	2.1	2146	256	104.49	105.51	0.9	3588
				2611.86		**710**			**3230.55**		**972**			**3042.33**		**1491**			**2591.67**		**2389**
Pla.	0.25	32	93.31	100.35	4.4	816	50	82.45	91.29	6.9	991	51	77.85	92.04	9.1	996	47	81.92	92.65	9.6	950
	0.30	32	83.77	102.64	12.9	820	64	51.98	62.56	6.7	1243	77	45.09	56.51	8.8	1266	78	48.44	57.94	9.2	1264
	0.35	32	72.82	83.81	7.2	805	64	47.31	53.44	4.4	1302	128	30.27	35.20	3.6	1938	141	28.51	33.94	3.9	1896
	0.40	32	68.09	77.60	6.5	848	64	43.15	47.89	2.7	1318	128	27.94	29.24	1.1	2013	256	18.56	19.00	0.4	3156
	0.45	32	67.30	76.41	8.3	894	64	40.16	43.22	2.1	1278	128	26.53	27.53	0.8	2017	256	16.68	17.24	0.3	3593
	0.50	32	66.99	72.90	6.3	880	64	40.95	43.98	2.4	1328	128	26.08	26.66	0.5	2061	256	16.24	16.54	0.2	3624
				85.62		**844**			**57.06**		**1243**			**44.53**		**1715**			**39.55**		**2414**
Blo.	0.25	32	97.30	114.61	10.5	775	52	73.09	91.74	9.3	923	55	75.03	89.37	12.7	958	51	78.16	96.12	14.5	916
	0.30	32	91.30	102.14	9.1	825	64	61.91	66.58	3.8	1174	102	50.15	56.18	4.6	1434	98	52.46	58.63	5.9	1413
	0.35	32	89.53	97.44	5.7	805	64	55.89	59.21	2.1	1257	128	38.25	42.92	3.1	1923	172	34.85	40.37	5.4	2177
	0.40	32	86.19	92.57	7.1	827	64	53.20	56.86	2.3	1259	128	35.72	38.40	1.9	2012	256	25.04	26.35	0.7	3231
	0.45	32	87.26	90.58	2.9	852	64	52.03	54.68	2.1	1250	128	33.08	34.85	1.0	2068	256	22.53	23.65	0.6	3320
	0.50	32	86.53	92.11	9.8	847	64	51.47	53.32	1.6	1243	128	32.53	33.53	1.0	2032	256	20.91	21.53	0.4	3399
				98.24		**822**			**63.73**		**1184**			**49.21**		**1738**			**44.44**		**2409**
Lak.	0.25	4	226.82	12532.80	4081.4	401	5	208.16	12531.10	4087.1	404	27	191.23	5148.80	6829.9	705	9	190.99	11295.95	5489.7	485
	0.30	18	226.50	6381.04	7068.1	616	1	13763.40	13763.40	0.0	348	34	100.60	10038.16	6380.2	736	26	98.09	11278.85	5527.8	621
	0.35	18	213.76	6378.91	7070.1	621	18	147.07	10050.02	6359.9	575	47	93.31	8793.50	6895.4	958	45	66.51	11273.48	5539.7	797
	0.40	9	212.37	10070.12	6325.5	473	35	140.95	6336.14	7111.1	831	59	92.29	7549.41	7139.3	1121	94	61.24	8780.92	6912.9	1478
	0.45	32	208.39	217.60	5.9	834	64	136.90	146.26	5.9	1250	128	91.13	92.82	1.1	2045	256	59.39	60.43	0.6	3648
	0.50	32	209.98	219.60	7.6	846	64	139.68	143.07	3.2	1299	128	90.26	91.88	1.1	2018	256	57.99	59.24	0.8	3608
				5966.68		**632**			**7161.67**		**784**			**5285.76**		**1264**			**7124.81**		**1773**

[10] G. Joy, Z. Xiang, Center-cut for color-image quantization, Vis. Comput. 10 (1) (1993) 62–66, http://dx.doi.org/10.1007/BF01905532.

[11] S.-C. Cheng, C.-K. Yang, A fast and novel technique for color quantization using reduction of color space dimensionality, Pattern Recognit. Lett. 22 (8) (2001) 845–856, http://dx.doi.org/10.1016/S0167-8655(01)00025-3.

[12] Y. Sirisathitkul, S. Auwatanamongkol, B. Uyyanonvara, Color image quantization using distances between adjacent colors along the color axis with highest color variance, Pattern Recognit. Lett. 25 (9) (2004) 1025–1043, http://dx.doi.org/10.1016/j.patrec.2004.02.012.

[13] K. Kanjanawanishkul, B. Uyyanonvara, Novel fast color reduction algorithm for time-constrained applications, J. Vis. Commun. Image Represent. 16 (3) (2005) 311–332, http://dx.doi.org/10.1016/j.jvcir.2004.07.002.

[14] M.E. Celebi, Q. Wen, S. Hwang, An effective real-time color quantization method based on divisive hierarchical clustering, J. Real-Time Image Process. 10 (2) (2015) 329–344, http://dx.doi.org/10.1007/s11554-012-0291-4.

[15] H. Kasuga, H. Yamamoto, M. Okamoto, Color quantization using the fast K-means algorithm, Syst. Comput. Japan 31 (8) (2000) 33–40, http://dx.doi.org/10.1002/1520-684X(200007)31:8<33::AID-SCJ4>3.0.CO;2-C.

[16] Y.-C. Hu, B.-H. Su, Accelerated k-means clustering algorithm for colour image quantization, Imaging Sci. J. 56 (1) (2008) 29–40, http://dx.doi.org/10.1179/174313107X176298.

[17] M.E. Celebi, Fast color quantization using weighted sort-means clustering, J. Opt. Soc. Amer. A 26 (11) (2009) 2434–2443, http://dx.doi.org/10.1364/JOSAA.26.002434.

[18] M.E. Celebi, Improving the performance of k-means for color quantization, Image Vis. Comput. 29 (4) (2011) 260–271, http://dx.doi.org/10.1016/j.imavis.2010.10.002.

[19] A.H. Dekker, Kohonen neural networks for optimal colour quantization, Network: Comput. Neural Syst. 5 (3) (1994) 351–367, http://dx.doi.org/10.1088/0954-898X/5/3/003.

[20] N. Papamarkos, A.E. Atsalakis, C.P. Strouthopoulos, Adaptive color reduction, IEEE Trans. Syst. Man Cybern. B 32 (1) (2002) 44–56, http://dx.doi.org/10.1109/3477.979959.

[21] C.-H. Chang, P. Xu, R. Xiao, T. Srikanthan, New adaptive color quantization method based on self-organizing maps, IEEE Trans. Neural Netw. 16 (1) (2005) 237–249, http://dx.doi.org/10.1109/TNN.2004.836543.

[22] A. Atsalakis, N. Papamarkos, Color reduction and estimation of the number of dominant colors by using a self-growing and self-organized neural gas, Eng. Appl. Artif. Intell. 19 (7) (2006) 769–786, http://dx.doi.org/10.1016/j.engappai.2006.05.004.

[23] C.-H. Wang, C.-N. Lee, C.-H. Hsieh, Sample-size adaptive self-organization map for color images quantization, Pattern Recognit. Lett. 28 (13) (2007) 1616–1629, http://dx.doi.org/10.1016/j.patrec.2007.04.005.

[24] E.J. Palomo, E. Domínguez, Hierarchical color quantization based on self-organization, J. Math. Imaging Vision 49 (1) (2014) 1–19, http://dx.doi.org/10.1007/s10851-013-0433-8.

[25] M.G. Omran, A.P. Engelbrecht, A. Salman, A color image quantization algorithm based on particle swarm optimization, Inform. (Slovenia) 29 (3) (2005) 261–270.

[26] J. Kennedy, Particle swarm optimization, in: Encyclopedia of Machine Learning, Springer, 2011, pp. 760–766.

Table 7
MSE (*min*: minimum, *av*: average, *dev*: standard deviation) and average time (milliseconds) for ITATCQ that uses constant α and random pixels — iteration 10 (q: number of colors of the quantized palette; Q_{max}: limit to the number of colors of the quantized palette).

		$Q_{max}=32$					$Q_{max}=64$					$Q_{max}=128$					$Q_{max}=256$				
	α	q	*min*	*av*	*dev*	T	q	*min*	*av*	*dev*	T	q	*min*	*av*	*dev*	T	q	*min*	*av*	*dev*	T
Sai.	0.25	32	81.27	104.03	14.3	1274	63	47.43	54.78	6.4	1724	70	46.43	71.82	57.4	1771	74	45.41	56.51	14.4	1839
	0.30	32	78.83	84.26	4.5	1279	64	42.79	46.43	2.2	1933	128	25.66	30.71	3.9	2905	159	22.83	28.86	3.9	3102
	0.35	32	72.42	82.30	8.2	1355	64	40.26	43.73	2.5	1971	128	23.56	25.08	1.1	3144	256	14.32	15.83	1.0	4877
	0.40	32	71.67	74.69	2.7	1396	64	37.67	40.35	1.7	1915	128	22.60	23.27	0.7	3127	256	13.56	14.08	0.3	5097
	0.45	32	66.27	73.62	3.8	1304	64	37.00	38.78	1.6	1887	128	21.70	22.22	0.4	3080	256	12.96	13.37	0.3	5117
	0.50	32	68.16	74.04	4.7	1315	64	36.57	38.18	1.2	1881	128	21.32	21.95	0.4	3044	256	12.62	12.99	0.2	5200
				82.16		**1320**			**43.71**		**1885**			**32.51**		**2845**			**23.61**		**4205**
Mot.	0.25	32	255.85	330.78	37.4	1238	41	200.26	294.55	47.1	1320	42	233.93	296.35	34.8	1342	39	237.93	317.75	34.0	1299
	0.30	32	207.78	228.91	13.5	1315	64	126.03	138.14	8.2	1913	124	74.64	92.40	21.6	2777	139	65.63	86.00	14.0	2780
	0.35	32	201.22	219.30	11.9	1322	64	124.09	132.15	6.1	2009	128	70.19	73.08	1.9	3212	256	46.01	46.96	0.7	5074
	0.40	32	204.58	214.91	11.4	1345	64	118.25	124.27	5.5	1990	128	68.57	71.13	1.5	3170	256	42.49	43.77	0.7	5468
	0.45	32	200.91	211.31	6.5	1364	64	112.45	116.52	3.1	1944	128	65.63	68.51	1.8	3131	256	41.15	41.91	0.7	5331
	0.50	32	195.45	207.56	6.7	1326	64	110.37	116.14	2.2	1906	128	65.01	66.91	0.9	3126	256	40.23	40.82	0.5	5311
				235.46		**1318**			**153.63**		**1847**			**111.40**		**2793**			**96.20**		**4211**
Lad.	0.25	31	138.55	215.40	39.9	1189	49	85.47	175.61	95.8	1394	51	78.79	170.70	85.2	1392	51	78.34	167.67	105.4	1386
	0.30	32	126.72	144.52	11.1	1316	64	69.56	76.82	4.2	1901	110	47.52	58.08	6.6	2468	109	51.11	57.06	6.0	2484
	0.35	32	125.34	131.41	5.3	1399	64	66.72	69.75	2.3	2031	128	41.60	44.38	1.8	3082	183	34.18	37.16	2.8	3764
	0.40	32	119.95	126.88	4.9	1412	64	65.79	67.42	1.4	2008	128	38.64	39.89	0.6	3098	256	25.02	25.91	0.5	5107
	0.45	32	118.70	126.39	6.3	1337	64	64.02	65.89	1.8	1948	128	37.09	37.69	0.5	3126	256	22.52	23.35	0.6	5289
	0.50	32	119.92	126.34	4.6	1298	64	63.45	66.10	3.1	1905	128	35.40	36.62	0.9	3081	256	21.22	22.02	0.4	5329
				145.16		**1325**			**86.93**		**1865**			**64.56**		**2708**			**55.53**		**3893**
Cap.	0.25	32	177.36	215.65	55.9	1290	64	92.41	104.94	11.2	1722	103	55.86	82.87	27.0	2220	99	68.87	88.59	20.3	2023
	0.30	32	169.49	195.54	18.4	1305	64	88.81	96.54	4.7	1957	128	49.88	53.54	2.6	3129	196	37.80	43.55	3.8	3670
	0.35	32	168.10	181.12	18.9	1384	64	83.24	93.81	6.9	1989	128	46.73	51.57	4.5	3204	256	28.62	29.77	1.0	5107
	0.40	32	161.16	177.62	15.9	1385	64	85.56	90.97	5.1	1953	128	44.23	47.16	1.8	3086	256	24.68	26.12	1.1	5296
	0.45	32	156.63	172.38	11.4	1303	64	75.54	82.94	7.1	1926	128	42.06	44.65	1.7	3086	256	23.38	24.49	0.6	5261
	0.50	32	161.42	172.02	11.2	1275	64	79.07	85.79	4.8	1905	128	40.91	42.79	1.7	3085	256	22.33	22.96	0.6	5280
				185.72		**1324**			**92.50**		**1909**			**53.76**		**2968**			**39.25**		**4439**
Par.	0.25	32	275.19	312.09	28.7	1255	49	192.85	258.22	36.5	1460	51	209.66	251.53	28.5	1494	49	220.39	250.97	20.3	1465
	0.30	32	267.89	280.03	12.3	1296	64	155.47	164.69	6.9	1899	99	118.25	132.89	7.8	2294	90	129.35	146.12	8.5	2145
	0.35	32	262.28	274.42	8.8	1335	64	144.14	153.12	5.0	2003	128	90.42	96.22	3.8	3049	227	64.15	72.94	6.9	4127
	0.40	32	242.47	261.77	10.6	1356	64	141.32	151.00	9.2	2001	128	82.21	85.95	2.6	3097	256	50.95	52.88	1.5	5210
	0.45	32	242.19	259.97	16.2	1342	64	134.15	148.13	7.6	1931	128	78.37	83.91	3.4	3095	256	47.70	49.53	1.9	5364
	0.50	32	242.62	262.31	18.9	1325	64	130.92	142.38	5.9	1874	128	77.64	83.82	4.5	3057	256	45.69	47.43	1.1	5456
				275.10		**1318**			**169.59**		**1861**			**122.39**		**2681**			**103.31**		**3961**
Gir.	0.25	10	276.55	14274.86	9629.2	782	26	179.25	8261.18	10322.9	1045	22	121.64	12240.21	10347.5	951	23	121.77	12232.00	10357.9	971
	0.30	20	148.88	8196.68	10378.3	1037	33	83.72	10174.38	10626.0	1263	65	52.86	10156.29	10645.1	1759	104	39.78	8129.92	10435.7	2204
	0.35	26	138.76	4173.27	8475.9	1209	51	77.26	4119.88	8504.0	1733	103	46.58	4089.84	8519.9	2643	180	29.47	6098.22	9769.2	3795
	0.40	32	129.26	141.37	6.9	1355	64	76.20	81.43	5.0	1966	128	43.48	46.37	1.6	3160	256	26.96	27.72	0.6	5293
	0.45	32	134.02	141.92	7.0	1338	64	74.72	79.95	5.8	1946	128	43.66	45.39	1.1	3087	256	26.02	26.75	0.5	5333
	0.50	32	131.53	140.25	6.1	1323	64	73.11	80.83	4.5	1928	128	43.30	44.59	1.5	3094	256	25.46	25.99	0.4	5368
				4511.39		**1174**			**3799.61**		**1647**			**4437.11**		**2449**			**4423.43**		**3827**

[27] C. Ozturk, E. Hancer, D. Karaboga, Color image quantization: a short review and an application with artificial bee colony algorithm, Inform 25 (3) (2014) 485–503, http://dx.doi.org/10.15388/Informatica.2014.25.

[28] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, J. Global Optim. 39 (3) (2007) 459–471, http://dx.doi.org/10.1007/s10898-007-9149-x.

[29] H. Azzag, G. Venturini, A. Oliver, C. Guinot, A hierarchical ant based clustering algorithm and its use in three real-world applications, European J. Oper. Res. 179 (3) (2007) 906–922, http://dx.doi.org/10.1016/j.ejor.2005.03.062.

[30] X.-S. Yang, X. He, Firefly algorithm: recent advances and applications, Int. J. Swarm Intell. 1 (1) (2013) 36–50, http://dx.doi.org/10.1504/IJSI.2013.055801.

[31] M.-L. Pérez-Delgado, Artificial ants and fireflies can perform colour quantisation, Appl. Soft Comput. 73 (2018) 153–177, http://dx.doi.org/10.1016/j.asoc.2018.08.018.

[32] M.-L. Pérez-Delgado, The color quantization problem solved by swarm-based operations, Appl. Intell. (2019) 2482–2514, http://dx.doi.org/10.1007/s10489-018-1389-6.

[33] M.-L. Pérez-Delgado, J.-Á. Román Gallego, A two-stage method to improve the quality of quantized images, J. Real-Time Image Proc. 17 (3) (2020) 581–605, http://dx.doi.org/10.1007/s11554-018-0814-8.

[34] M.-L. Pérez-Delgado, Color image quantization using the shuffled-frog leaping algorithm, Eng. Appl. Artif. Intell. 79 (2019) 142–158, http://dx.doi.org/10.1016/j.engappai.2019.01.002.

[35] P. Scheunders, A genetic c-means clustering algorithm applied to color image quantization, Pattern Recognit. 30 (6) (1997) 859–866, http://dx.doi.org/10.1016/S0031-3203(96)00131-8.

[36] D. Özdemir, L. Akarun, A fuzzy algorithm for color quantization of images, Pattern Recognit. 35 (8) (2002) 1785–1791, http://dx.doi.org/10.1016/S0031-3203(01)00170-4.

[37] G. Schaefer, H. Zhou, Fuzzy clustering for colour reduction in images, Telecommun. Syst. 40 (1) (2009) 17–25, http://dx.doi.org/10.1007/s11235-008-9143-8.

[38] 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.

[39] Q. Wen, M.E. Celebi, Hard versus fuzzy c-means clustering for color quantization, EURASIP J. Adv. Signal Process. 2011 (1) (2011) 1–12, http://dx.doi.org/10.1186/1687-6180-2011-118.

[40] M.E. Celebi, S. Hwang, Q. Wen, Color quantization using the adaptive distributing units algorithmic, Imaging Sci. J. 62 (2) (2014) 80–91, http://dx.doi.org/10.1179/1743131X13Y.0000000059.

[41] A. Weber, USC-SIPI image database, 2019, http://sipi.usc.edu/database/database.php?volume=misc/. (Accessed 5 June 2019).

[42] R. Franzen, Kodak lossless true color image suite, 2019, http://r0k.us/graphics/kodak/. (Accesed 5 June 2019).

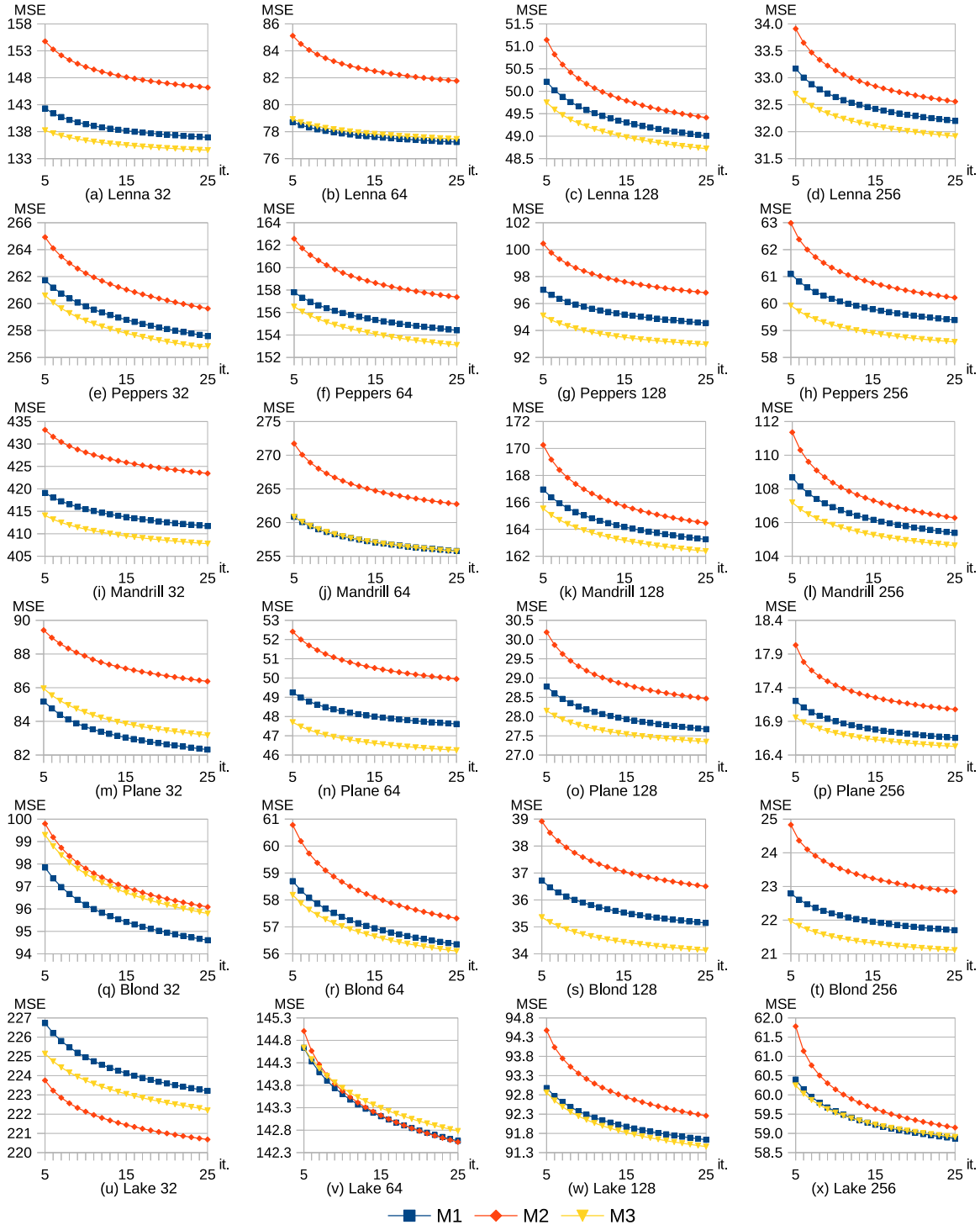


Fig. 6. Average MSE for ITATCQ-n with the increase of α applied with three different functions: M1: $0.2e^{-0.2it}$, M2: $0.2 - 0.2e^{-0.2it}$ M3: $(1 - \alpha_{ini})/3$, applied to the 4 initial iterations (it: iteration of the algorithm; α_{ini}: initial value of the α parameter).

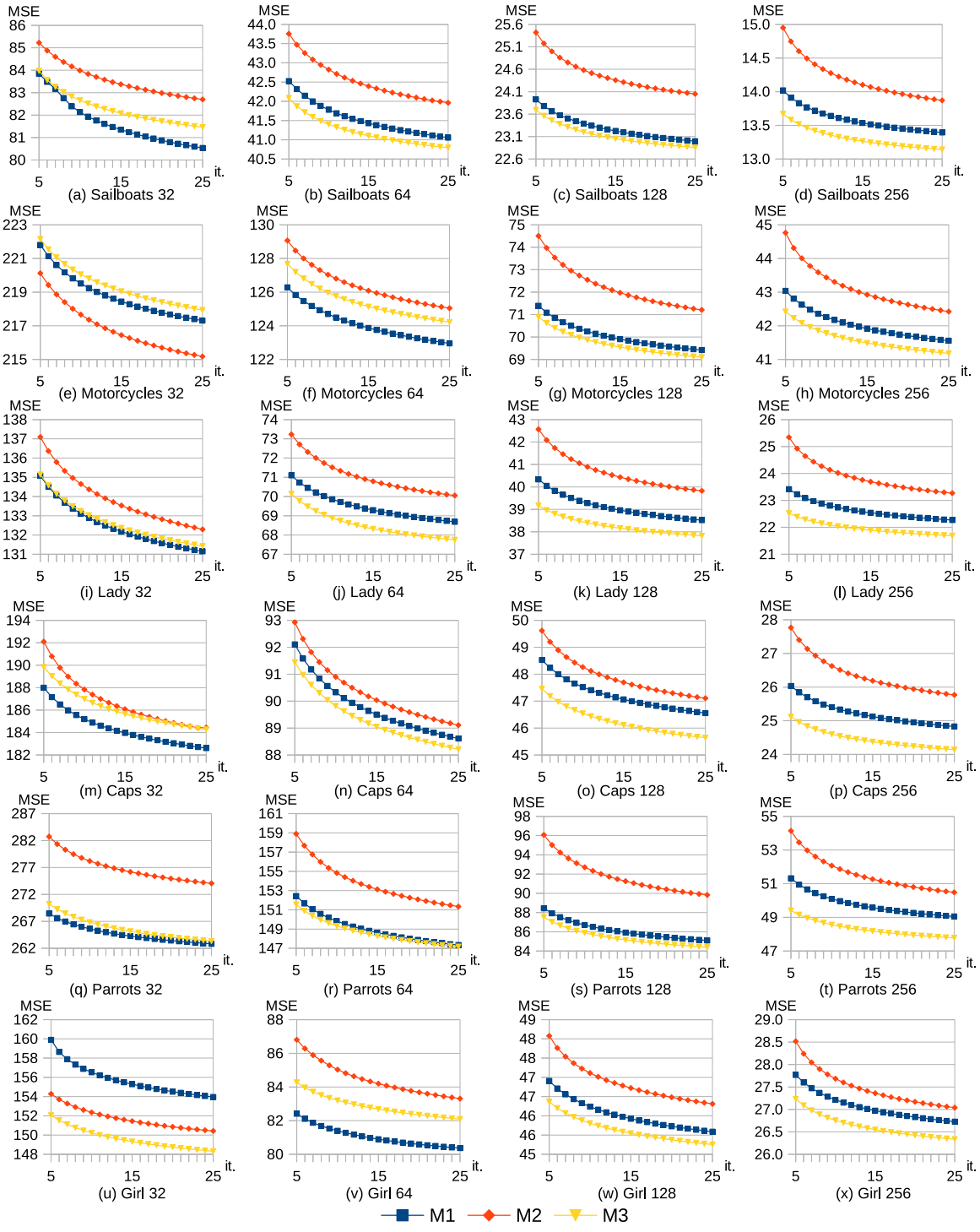


Fig. 7. Average MSE for ITATCQ-n with the increase of α applied with three different functions: M1: $0.2e^{-0.2it}$, M2: $0.2 - 0.2e^{-0.2it}$ M3: $(1 - \alpha_{ini})/3$, applied to the 4 initial iterations (it: iteration of the algorithm; α_{ini}: initial value of the α parameter).

Table 8
MSE (*min*: minimum, *av*: average, *dev*: standard deviation) and average time (milliseconds) for several color quantization methods (KM: K-means; SFLA: Shuffled-frog leaping algorithm; ABC+ATCQ: ABC with ATCQ; ATCQ+FA: Firefly algorithm with ATCQ).

		KM				SFLA				ABC+ATCQ				ATCQ+FA			
	colors	*min*	*av*	*dev*	*T*	*min*	*av*	*dev*	*T*	*min*	*av*	*dev*	*T*	*min*	*av*	*dev*	*T*
Lenna	32	123.80	131.36	6.8	2822	119.23	122.33	2.1	949	118.68	120.12	0.9	9689	137.87	139.40	1.8	4169
	64	79.06	81.59	1.8	5414	73.20	74.55	1.0	1825	72.37	73.22	0.7	17556	79.53	80.34	0.6	6840
	128	49.21	50.65	1.4	10736	47.33	48.27	0.7	3625	46.28	46.64	0.2	29840	50.35	50.92	0.6	11024
	256	32.40	33.14	0.4	22351	31.52	32.12	0.3	7426	30.34	30.44	0.1	56854	33.07	33.42	0.2	17923
Peppers	32	245.81	262.37	13.1	2837	228.51	236.01	3.7	871	227.96	232.09	1.9	10612	286.06	290.75	2.6	4070
	64	145.92	152.16	4.4	5521	134.70	141.65	3.8	1622	134.14	135.35	0.9	16971	160.17	161.05	0.8	7037
	128	87.68	94.18	4.9	10655	85.63	87.94	2.5	3179	83.62	84.16	0.3	29753	97.60	98.24	0.4	11657
	256	57.90	59.00	0.7	21316	56.49	57.49	0.7	6575	53.87	54.28	0.2	57838	62.22	63.21	0.6	17607
Mandrill	32	388.65	394.71	6.1	2837	376.23	383.71	6.1	901	375.43	378.57	2.0	11029	404.66	408.79	1.9	3783
	64	243.53	249.61	5.2	5506	239.24	241.35	1.6	1691	237.50	238.35	0.9	17703	261.56	264.80	2.3	6267
	128	157.51	158.77	1.4	10692	153.94	155.39	0.9	3285	152.22	153.20	0.4	30156	167.26	170.09	1.4	10353
	256	101.06	102.08	0.9	21138	100.02	101.12	0.6	6627	97.51	98.03	0.2	57805	110.35	110.72	0.3	16017
Plane	32	125.21	150.57	17.5	2498	81.16	106.83	15.9	895	71.03	74.58	2.5	10624	98.54	100.97	1.0	3964
	64	52.61	93.27	21.8	4901	49.27	58.41	5.1	1606	45.48	47.75	1.9	17412	47.12	47.97	0.5	6716
	128	37.82	41.63	2.4	9542	32.36	36.72	2.2	3087	29.32	30.56	0.7	30659	27.63	27.96	0.2	10324
	256	23.68	25.64	1.6	18793	22.63	24.35	1.3	6287	19.75	20.59	0.4	57659	17.96	18.21	0.2	15312
Blond	32	91.24	116.32	13.4	2861	84.92	90.21	5.7	882	81.81	82.69	0.5	10383	93.07	100.24	3.7	4193
	64	54.40	62.51	8.6	5460	51.64	52.98	0.7	1653	49.42	49.80	0.2	17951	58.76	59.43	0.6	7020
	128	35.31	37.63	2.4	10959	32.59	33.84	0.9	3281	29.86	31.09	0.6	32205	37.68	38.06	0.3	11600
	256	21.39	23.66	1.1	21434	20.85	22.36	0.8	6333	18.94	19.84	0.6	57458	21.85	22.11	0.2	16985
Lake	32	211.63	224.24	12.3	2814	203.54	208.17	2.4	931	204.91	206.51	1.3	10597	217.65	219.79	1.2	4144
	64	139.05	147.59	4.9	5475	132.70	135.93	2.6	1619	131.45	132.70	0.9	17680	153.63	155.03	1.2	7070
	128	91.36	95.59	2.8	10715	88.52	90.97	1.3	3061	85.11	86.07	0.6	31248	100.26	101.31	0.8	11850
	256	60.11	63.29	2.2	21245	59.36	60.54	0.7	6192	55.48	55.83	0.3	58748	63.35	63.67	0.3	18795
Sailb.	32	88.23	104.10	16.8	4250	81.83	88.68	4.9	1314	67.18	70.42	2.2	15608	75.33	76.57	0.7	6324
	64	46.17	63.27	14.2	8286	49.66	53.06	3.3	2612	36.09	37.43	0.8	26404	42.78	44.10	0.8	10676
	128	26.84	31.91	3.4	16045	29.53	32.42	1.7	5337	21.54	21.92	0.3	46665	25.28	25.62	0.2	18986
	256	17.83	19.38	1.4	32105	20.71	21.90	1.2	11593	12.78	13.40	0.3	84949	14.66	14.91	0.2	30839
Motor.	32	200.27	229.11	20.1	4196	194.72	200.99	5.1	1328	189.13	192.31	2.4	16159	231.45	232.90	0.7	6307
	64	117.97	125.07	7.9	8274	113.28	119.01	3.4	2418	107.74	109.63	1.2	26519	129.78	131.23	1.0	10774
	128	74.04	76.75	2.8	16003	68.25	72.05	2.3	4878	63.14	64.03	0.6	46672	79.24	80.24	0.6	18656
	256	44.22	48.19	1.8	32043	45.89	47.24	1.0	9886	38.14	38.67	0.4	86689	48.95	49.43	0.5	33075
Lady	32	118.27	134.40	11.0	4231	117.07	121.59	4.2	1380	113.74	115.79	1.2	15756	136.79	140.56	1.8	6158
	64	66.42	74.14	7.6	8186	65.91	67.57	1.3	2763	61.40	62.31	0.5	26491	71.61	72.78	0.9	10672
	128	37.93	40.39	1.8	16000	38.93	40.48	1.5	5673	33.83	34.30	0.4	44446	43.06	43.47	0.4	18195
	256	22.23	23.65	0.8	31915	23.10	24.71	0.9	11610	19.13	19.41	0.2	84386	24.19	24.45	0.3	29605
Caps	32	184.81	198.36	12.7	4313	160.78	185.48	15.3	1208	152.40	162.32	6.1	16167	224.09	226.02	1.3	5946
	64	99.75	111.78	7.4	8254	92.43	101.69	5.8	2512	76.54	79.53	2.2	25522	98.34	101.18	1.3	10618
	128	50.81	59.36	5.7	16086	47.86	54.40	5.4	5270	39.90	40.96	1.0	45160	52.28	53.60	0.7	18882
	256	29.54	33.43	2.2	31665	28.60	32.17	2.0	11683	21.97	23.28	0.7	84801	27.40	28.23	0.5	31339
Parrots	32	246.34	280.72	29.6	4265	237.55	245.33	8.1	1231	231.78	236.34	3.5	15912	269.13	270.94	1.3	6193
	64	139.75	149.05	6.5	8242	132.43	137.98	4.3	2543	127.32	128.92	1.1	26094	150.75	154.86	4.2	11067
	128	81.11	85.03	4.2	16182	78.38	80.22	1.1	5155	72.69	74.35	0.9	45012	86.76	89.79	1.4	17740
	256	48.64	50.25	1.2	31520	47.94	49.24	1.2	11507	42.65	43.27	0.4	85512	52.28	53.03	0.7	28567
Girl	32	141.79	159.20	13.3	4299	128.01	136.17	6.3	1289	125.78	128.42	1.6	16074	203.00	207.46	3.6	6083
	64	80.73	87.84	6.3	8200	76.91	79.38	2.2	2421	71.64	73.16	0.7	27068	102.26	103.76	1.3	10407
	128	52.07	56.22	6.1	16092	47.20	49.74	1.8	4698	41.01	41.56	0.4	45535	54.63	55.31	0.7	18790
	256	31.85	34.18	1.4	31694	30.53	32.01	0.9	10150	24.35	24.64	0.3	86619	29.22	30.55	0.7	31278

Table 9
MSE (*min*: minimum, *av*: average, *dev*: standard deviation) and average time (milliseconds) for several color quantization methods (BS: Binary splitting; WU: Wu's method; OC: Octree; VB: Variance-based method; MC: Median-cut; NQ: Neuquant; BS+ATCQ: Binary splitting with ATCQ; ATCQ: Ant-tree for color quantization).

		BS		WU		OC		VB		MC		NQ		BS+ATCQ		ATCQ			
	colors	MSE	T	MSE	T	MSE	T	MSE	T	MSE	T	MSE	T	MSE	T	*min*	*av*	*dev*	T
Lenna	32	138.44	171	158.61	12	482.03	174	203.07	230	425.85	12	152.13	139	123.82	246	163.55	208.99	45.9	66
	64	82.44	219	99.16	13	212.92	178	135.86	291	362.61	12	85.60	202	75.35	358	96.16	128.83	33.1	105
	128	53.04	267	61.79	13	140.53	183	94.68	332	276.33	13	53.73	308	48.10	561	58.87	93.05	47.2	158
	256	35.28	350	39.53	13	74.12	188	69.41	387	193.25	14	34.88	512	31.59	872	38.78	80.91	55.9	223
Peppers	32	311.66	173	279.27	12	777.14	168	451.10	297	866.55	13	283.69	154	254.22	246	334.17	421.08	79.9	62
	64	173.75	213	165.37	14	495.51	171	304.21	364	771.03	13	166.37	220	145.66	402	188.96	274.94	114.7	99
	128	106.85	277	102.31	13	308.76	189	212.68	427	585.41	14	95.69	324	91.29	543	114.89	224.60	152.5	161
	256	68.72	344	66.08	13	156.24	181	147.18	501	397.29	15	63.08	535	58.38	848	73.32	205.57	168.6	229
Mandrill	32	441.30	184	468.39	12	1094.11	172	531.03	320	984.05	14	456.13	142	397.18	261	538.49	2023.16	3501.7	60
	64	286.93	229	288.33	13	576.19	186	346.58	402	881.29	15	272.25	218	248.82	379	362.35	1913.97	3557.1	92
	128	183.51	284	186.33	13	357.13	178	248.02	476	776.65	16	168.22	316	160.71	562	220.56	1837.06	3597.3	133
	256	117.85	369	118.65	14	195.82	184	181.94	594	676.87	22	109.34	510	103.18	887	147.20	1797.99	3618.3	191
Plane	32	83.12	148	85.45	13	342.23	166	123.56	216	374.70	13	123.70	126	68.10	204	116.71	174.35	56.0	57
	64	46.22	201	51.33	13	225.98	172	80.73	246	237.97	12	57.57	176	39.30	327	64.15	101.61	43.0	88
	128	28.06	243	32.60	15	133.59	178	52.60	298	187.93	14	29.71	274	24.24	512	35.49	78.45	51.4	123
	256	18.42	321	21.66	14	52.31	183	36.85	357	147.53	14	21.09	473	15.77	839	24.62	66.65	58.6	157
Blond	32	111.28	172	107.55	13	477.64	172	232.43	132	374.69	12	123.89	136	95.20	247	132.69	177.37	41.4	65
	64	63.07	225	63.03	13	163.08	177	129.43	150	271.55	13	66.32	199	51.89	379	97.80	120.36	20.8	103
	128	38.46	276	36.84	13	89.86	184	82.22	171	185.81	15	38.30	302	32.04	557	55.38	83.72	36.5	150
	256	23.65	379	23.59	14	50.79	185	54.42	190	123.57	14	23.86	496	19.45	877	28.07	69.28	46.9	205
Lake	32	245.65	173	249.81	13	922.04	167	357.26	371	1208.87	13	274.45	151	218.92	234	268.93	482.77	285.8	66
	64	162.89	217	161.34	13	466.14	169	251.70	449	984.62	13	164.26	204	140.32	365	182.78	222.66	48.8	110
	128	107.00	268	102.55	13	198.49	186	170.97	556	692.56	15	100.88	314	90.70	543	122.82	165.14	70.8	173
	256	68.36	347	66.47	13	159.21	177	120.10	664	523.09	16	65.70	512	57.65	834	78.16	142.53	87.4	245
Sailb.	32	86.96	248	85.82	19	477.06	243	139.41	234	320.63	183	118.56	149	73.86	356	91.87	201.14	132.4	90
	64	47.36	278	50.30	20	144.83	260	71.01	292	200.83	180	59.26	245	39.67	494	56.43	79.74	32.1	148
	128	27.39	389	29.23	19	118.13	295	44.09	356	176.83	181	28.26	375	21.58	785	31.37	41.20	8.7	219
	256	14.99	508	18.89	20	41.05	360	27.62	413	126.59	187	18.00	658	12.37	1247	19.00	29.69	14.2	333
Motor.	32	276.14	258	268.42	19	849.84	256	406.60	266	532.58	183	339.71	177	224.66	374	276.33	371.08	107.9	101
	64	151.98	332	147.10	19	351.29	306	216.75	322	398.09	186	140.49	262	118.62	542	155.17	197.81	32.5	179
	128	83.79	422	86.71	19	230.56	343	119.36	386	299.41	180	79.60	438	68.55	815	97.87	124.88	46.1	299
	256	49.37	551	51.14	19	126.16	407	73.01	460	228.45	187	46.93	726	40.17	1313	57.63	93.74	62.2	485
Lady	32	144.81	259	148.83	18	412.51	240	417.87	362	317.46	177	160.77	170	116.20	364	184.97	229.60	37.4	97
	64	79.32	313	82.36	19	333.21	278	315.04	435	231.33	180	81.23	259	66.49	515	92.97	140.27	54.8	170
	128	44.57	390	46.69	19	126.93	318	189.51	524	168.37	183	41.66	420	36.78	799	52.58	75.43	19.3	260
	256	24.55	509	27.34	19	78.17	355	90.16	622	121.81	184	23.27	706	20.57	1287	29.67	50.46	22.8	365
Caps	32	217.36	215	213.01	18	782.08	210	352.56	254	637.58	181	231.86	170	177.84	327	300.38	523.68	155.4	100
	64	112.00	271	102.70	18	323.01	245	163.94	316	431.91	186	93.91	261	82.13	487	129.32	234.91	106.7	166
	128	55.43	352	51.90	19	171.52	281	75.92	384	305.15	178	49.04	415	40.95	748	75.38	119.33	65.0	254
	256	28.86	446	29.68	19	70.45	322	43.02	438	212.22	187	27.73	715	22.18	1199	40.25	65.34	25.8	387
Parrots	32	315.58	239	298.98	18	853.02	228	357.24	260	862.83	179	297.85	178	249.72	344	361.84	590.07	166.4	92
	64	163.83	291	167.24	18	364.27	263	217.62	317	609.94	184	156.24	294	133.34	493	240.89	330.44	82.0	162
	128	94.54	358	95.39	19	253.69	285	143.31	379	441.59	187	86.03	433	79.40	756	142.39	208.99	52.0	240
	256	54.36	468	58.43	20	123.65	343	98.34	449	311.14	184	50.41	732	45.27	1220	78.32	137.70	57.2	367
Girl	32	151.51	242	162.03	18	508.41	238	222.09	337	334.74	183	169.25	158	136.95	347	369.93	544.28	170.1	101
	64	89.28	294	93.38	19	328.70	262	119.52	429	258.61	180	93.44	260	77.10	499	136.90	197.15	64.4	167
	128	52.19	371	54.95	20	169.86	300	80.14	504	200.69	179	51.39	395	43.26	763	68.34	108.18	26.9	286
	256	30.06	493	32.96	20	109.06	337	53.84	581	159.37	182	30.80	705	24.87	1266	40.88	63.07	18.7	438