Available online at www.sciencedirect.com

ScienceDirect

Pattern Recognition Letters 28 (2007) 1383–1390

Pattern Recognition Letters

www.elsevier.com/locate/patrec

Color reduction based on ant colony

Avazeh Tashakkori Ghanbarian [*], Ehsanollah Kabir, Nasrollah Moghaddam Charkari

Department of Electrical Engineering, Tarbiat Modarres University, P.O. Box 14115-143, Tehran, Iran

Received 10 January 2006; received in revised form 9 December 2006
Available online 28 February 2007

Communicated by A.M. Alimi

Abstract

In this article a method for color reduction based on ant colony algorithm is presented. Generally color reduction involves two steps: choosing a proper palette and mapping colors to this palette. This article is about the first step. Using ant colony algorithm, pixel clusters are formed based on their colors and neighborhood information to make the final palette. A comparison between the results of the proposed method and some other methods is presented. There are some parameters in the proposed method which can be set based on user needs and priorities. This increases the flexibility of the method.

Keywords: Color reduction; Clustering; Ant colony; Optimum palette; Image processing

1. Introduction

Researchers in the field of artificial intelligence have been inspired by the nature for many years. We can point to neural networks, evolutionary algorithms and artificial immune systems as well-known examples. Also some algorithms have been introduced based on swarm behavior studies (Dorigo et al., 1996; Deneubourg et al., 1990). Ant colony optimization, ACO (Dorigo et al., 1996), is a metaheuristic proven to be successful in solving NP-hard optimization problems. ACO is a distributed solution which uses pheromones as a positive feedback to enable the ants to communicate. On the other hand, ant-based clustering uses positive feedback and local information. It should be considered that ACO and ant-based clustering have many differences. The most important difference is that pheromones are not used in ant-based clustering.

The first clustering method based on ant colony was introduced by Deneubourg et al. (1990). They modeled ants as simple agents which move randomly in a square grid. Data items are scattered randomly on this grid and can be picked up, moved and dropped by ants. In this model ants are likely to pick up the items which are surrounded by dissimilar items and have tendency toward dropping them near similar items. Picking up and dropping down possibilities are calculated through simple formulas. By iterating these actions, the distribution of items on the grid will change. This distribution is used as a feedback and by repeating these operations, the items are clustered on the grid.

Several studies proceeded different aspects of Deneubourg's model like straight movements. Lumer and Faieta (1994) expanded it for data analysis. They presented a modified method on numeric data and improved convergence time. They described data items through numeric vectors and used Euclidean distance to calculate the distance between them. Generally this algorithm can be used on any data set to which a function can be declared as a measurement of dissimilarity. Also they deployed a short-term memory to memorize last transmitted items and the position they were dropped. After picking up an item, the ant identifies the most similar item in its memory and move

[*] Corresponding author. Fax: +98 21 88005040.
E-mail address: a_t_ghanbarian@hotmail.com (A.T. Ghanbarian).

0167-8655/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2007.01.019

towards its position. Indeed they introduced the heterogeneous populations of agents for the first time which are the agents with different individual initialized parameters.

After Lumer and Faieta, Kuntz et al. proposed a method using ant-based clustering for graph partitioning (Kuntz and Snyers, 1994; Kuntz and Snyers, 1999; Kuntz et al., 1998). Also other studies proceeded ant-based clustering applicability in document retrieval and visualization. In (Handl et al., 2003; Handl and Meyer, 2002) Deneunbourg's basic algorithm was deployed on web page categorization and visualization. Handl and Meyer used ant-based clustering to create topic maps dynamically (Handl and Meyer, 2002). Handl's approach starts with randomly scattering the data items on a grid. In the next phase, each ant randomly selects a data item from the grid and then the ant is randomly placed on one of the empty cells on the grid. By the end of this phase each ant has picked up an item so the main loop starts. In the main loop, at first an ant is randomly selected then it moves one step of a given step size. The ant probabilistically decides whether to drop its carried data item. If it drops the item, it immediately searches for a free item to pick up probabilistically. Then the main loop is repeated for another randomly selected ant. More detailed information on Handl's approach can be found in (Handl et al., 2003; Handl, 2003).

Other approaches combine ant colony with other methods like C-means. The aim of this hybridization is to cover the disadvantages of one method with the advantages of the other. Ant-based clustering makes initial clusters for C-means instead of having no primary information. On the other hand C-means easily handles misplaced or free items and improves quality (Steinberg et al., 1998; Monmarché, 1999). In another approach real ants' behavior is simulated by a hybridization of ant-based clustering and fuzzy if-then rules (Kanade and Hall, 2003). Labroche et al. (2003) proposes a new model based on ants' chemical recognition system to cope with unsupervised clustering problems.

Image color reduction or quantization is one of the basic image processing techniques. Color reduction has two main steps: (1) K colors are chosen from the original ones to form the color palette. (2) The output image is reconstructed using this palette. Obviously quality of the output image depends on the first step, palette design. Palette design methods are divided into two main categories: (1) Palette design with pre-determined number of colors. (2) Designing the best palette without specifying the number of its colors. Many techniques have been proposed in each category (Scheunders, 1997; Papamarkos et al., 2002; Bing et al., 2004; Dekker, 1994).

In this article a method for palette design with pre-determined number of colors using ant-based clustering is introduced (Ghanbarian and Kabir, 2006). The rest of the paper is organized as follows. Section 2 presents the principal characteristics of the proposed method and also introduces the color reduction algorithm. A discussion on parameter analysis and experimental results on our method and five other methods is presented in Section 3. Finally Section 4 draws the conclusion.

2. The proposed method

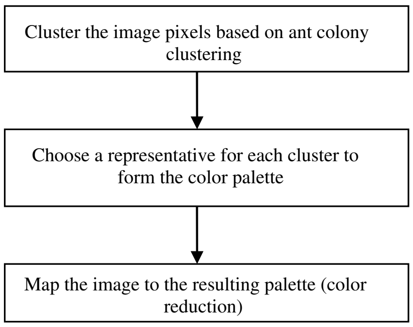
In this section the general arrangement, the block diagram and the algorithm of the proposed method is introduced. Fig. 1 shows the block diagram of the proposed method. In the first phase, a modified ant-based clustering algorithm is applied to image pixels. The second phase is dedicated to choosing a representative for each cluster. In the third phase, pixels are mapped to the formed palette and color reduction is completed. The simplest method is used for mapping pixels to the palette. In this method closest item to each pixel in the palette replace it in the final reconstructed image. Modified ant-based clustering algorithm is discussed in the following.

2.1. Ant-based color clustering

This method is based on the algorithm presented by Handl (Handl and Meyer, 2002). Handl's algorithm has been proposed for data clustering applications. In these applications, the data items are randomly spread on a grid. However in the proposed method, for color clustering problem, the algorithm is directly applied on the image grid itself. In this way the color and adjacency information are considered simultaneously in the clustering process. The neighbourhood function, $f(i)$, used in our method is as follows:

$$f(i) = \max\left(0, \frac{1}{\delta^2}\sum_j \left(1 - \frac{d(i,j)}{\alpha}\right)\right) \tag{1}$$

$d(i,j)$ defines the dissimilarity between pixels i and j. Parameter α determines the influence of dissimilarity function on $f(i)$. δ is the radius of perception. In this method, δ is set to 1 so that 8 pixels around each pixel are considered in calculating $f(i)$. The experiments showed that larger radius of perception extends the execution time while has very low impact on the quality of the results.


Cluster the image pixels based on ant colony clustering
Choose a representative for each cluster to form the color palette
Map the image to the resulting palette (color reduction)
Fig. 1. The proposed color reduction method.

```
1- Initialize ant parameters
2- For #iteration do
3- Pick up a random pixel
4- Check memory for similar item
5- If similar item exists then update the memory and drop the item
6-else repeat
        (a) Move to the best match neighbor
        (b) If f(i) > DTh then drop pixel
    until item has been dropped or K moves has been done
7-End for
```

Fig. 2. Proposed algorithm.

Some neighbours to i may be empty since their pixels have been picked up previously. Consequently less items are involved in calculating $f(i)$, thus $f(i)$ is smaller in sparse areas than it is in dense areas. Likewise similarity in dense areas is more than the similarity in sparse areas; therefore ants prefer to drop their items in dense areas rather than in sparse areas. This leads to condensing the areas.

The main algorithm is presented in Fig. 2. First step includes initialization of the parameters. These parameters are discussed in details later. Then the algorithm's main loop starts. In the beginning of each iteration, ant picks up an item randomly, then it searches its memory to find a similar item. If it finds such an item, it will drop its item in the position stored in its memory. Also it will update its memory. Otherwise it will start the inner loop. This loop starts with taking a step towards the best neighbour. Then neighbourhood function, $f(i)$, is calculated for the carried pixel according to the new position and its value is compared to the drop threshold, DTh. If it is greater than DTh, the current position is suitable for dropping the carried pixel, so ant drops the pixel. Otherwise ant keeps on performing the inner loop instructions. This is repeated until ant drops the carried pixel on a better point or it passes K steps, in this case the carried pixel is deleted. Finally the palette is formed regarding the items which exist in ant's memory. Picking up and dropping stages and their effects on resulting images' quality is explained in more details in the following sections. Also ant's memory and palette forming is discussed.

2.1.1. Pixel picking

In Handl's algorithm clusters can be recognized due to the dense homogeneous population of the items on the grid; so just when $f(i)$ is less than the picking up threshold, which means it is in a heterogeneous area, the item will be picked up. In our method, unlike Handl's method, which allows just one item per cell, there can be more than one item in one cell. In other words in this method each cluster is created by gathering data items in a cell, making a heap. The idea of heaps has also been used in some ant-based clustering approaches (Monmarché, 1999; Kanade and Hall, 2003). Using heap will lead to simplicity in dropping items, spatial discrimination of clusters and simpler cluster recognition at the end of the algorithm (Fig. 3). As a consequence random picking will be used not only to increase the variety of the pixels seen by the ant but also to decrease the execution time.

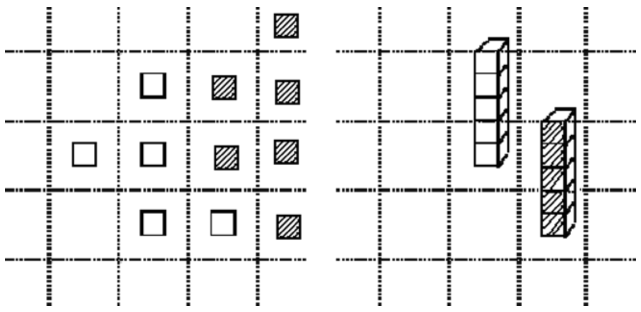


Fig. 3. Two clusters made in Handl method are shown on the left. The items of two clusters locating beside each other can be hardly recognized. Two clusters created by the proposed method are shown in the right.

2.1.2. Ant movement

Another difference between the proposed method and Handl algorithm is the way ant moves. In our algorithm, ant moves to its most similar neighbour, which is a pixel with the lowest $d(i,j)$ in ant's current neighbours. This method has been chosen due to meaningfulness of pixel neighbourhood in the image. This information is useful in color clustering in a way that the ant is likely to find more similar pixels in its next new neighbourhood. In addition the ant marks each point it passes not to examine it again.

2.1.3. Pixel dropping

Pixel dropping is performed in the inner loop; this loop is iterated until the pixel is dropped or ant has passed K steps. In our method when the ant wants to drop the item, it simply adds it to the heap. This leads to faster execution and more spatial discrimination of clusters (Fig. 3). In the case that the picked up pixel is a noisy pixel, there would be no similar item to it or in the case that the ant has chosen a wrong path and it cannot reach a suitable vicinity to find an appropriate match for the item, the ant leaves the pixel not to waste processing resources; this is the case when after K steps, ant leaves the item and finds another pixel to carry in order to be able to examine more items in less time.

2.1.4. Ant's memory

In this method ants can remember a number of the last carried pixels and their drop positions. One advantage of using memory is that the pixels are condensed faster to form the initial clusters. After picking up a pixel, ant refers to its memory searching for the best match for the current pixel picked up. The best match for pixel i is the one with the least dissimilarity to it. If dissimilarity is less than the memory threshold, MTh, ant will directly drop the pixel in place of the best match and updates its memory. Otherwise it will try to drop it using the ordinary drop step described above. At the end, if it can find an appropriate place for the pixel, it will drop the pixel and adds the pixel and its dropping place to its memory for future use.

Memory size is assumed to be fixed. Ant's memory size is set equal to the palette length. If an ant wants to add a new item to its memory and its memory is full, then it will replace the one with the lowest dropping number. At the end, when the ant repeats the outer loop sufficiently (Fig. 2), its memory is filled with the information needed to form the palette. Simply we will use the heaps' representatives which are in the ant's memory to form the palette.

3. Experimental results

As described before, the proposed algorithm starts with random picking of the image pixels which is simply a sampling phase. The result is directly affected by this phase. We used large number of pixels in all samples to avoid oscillating result due to different images. Furthermore in order to increase the precision and reliability, the average of some consecutive results is used in analyses. Four images which have been used for analysis are shown in Fig. 11. Three of these images are from USC[1] database and the last one, Flower image, is an arbitrary image chosen as a representative of images with large number of colors. In diagrams demonstrated in this section, reduced size images have been used because average result of at least 5 runs should be calculated for each point shown in diagrams (ACDSee 3.1 with lanczose filter with highest quality has been deployed for size reduction of images to 256×256). However reported AQE is very close to AQE of the original size image in every case.

The test platform was a Pentium III 800 MHz with 256 MB RAM. The algorithm was implemented by Visual Basic 6.0. For statistical comparison we used AQE[2] as a measure for the amount of distortion in the quantized image. If $A(i,j,k)$ is kth color element of (i,j) pixel of original image and $B(i,j,k)$ is the corresponding color element in the quantized image, then AQE is defined as follows:

$$\mathrm{AQE} = \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} \sqrt{\sum_{k=1}^{3} (A(i,j,k) - B(i,j,k))^2} \tag{2}$$

The rest of this section includes some experiments on distance function, parameters analysis and comparison between the proposed method and some other methods.

3.1. Distance function

As it was described in Section 2, $d(i,j)$ is a distance function defining dissimilarity between pixels i and j. L1, Euclidean and Chebyshev distance functions were tested (Eqs. (3)–(5)). Tested over many images, it was hard to find a suitable threshold, DTh, in our algorithm (Fig. 2).

Table 1
Comparison of L1 and Euclidean distances based on AQE

	Lenna	Baboon	Pepper	Flower
L1 distance	8.84	11.16	9.50	12.19
Euclidean distance	12.05	12.82	10.61	12.19

$$d(i,j) = |R_i - R_j| + |G_i - G_j| + |B_i - B_j| \tag{3}$$

$$d(i,j) = \sqrt{(R_i - R_j)^2 + (G_i - G_j)^2 + (B_i - B_j)^2} \tag{4}$$

$$d(i,j) = \max\left(|R_i - R_j|, |G_i - G_j|, |B_i - B_j|\right) \tag{5}$$

Comparing to Chebyshev distance, Euclidean distance, Eq. (5), yields better quantized images, but it is still behind L1 distance in producing images with less distortion. Table 1 lists AQE of average results over 10 runs for L1 and Euclidean distances. Algorithm's parameters were set to $\alpha = 1$, $\mathrm{DTh} = 0.6$, $\mathrm{MTh} = 35$, $X = 20$ and Y=10(Eqs. 6 and 7). More comments on parameter setting is included in Section 3.2. Based on these experiments L1 distance was used in the algorithm.

3.2. Parameter analysis

This subsection is dedicated to analysing the parameters used in our algorithm.

3.2.1. Parameter α

Parameter α in Eq. (2), determines the influence of dissimilarity function on $f(i)$. Mean results of 5 runs are shown in Figs. 4 and 5. Other parameters are $\mathrm{MTh} = 35$ and $\mathrm{DTh} = 0.6$ and ant memory size is 64. When α increases, the number of steps ant passes before dropping the item is reduced. This is because the dissimilarity influence on $f(i)$ is reduced and the ant is more likely to find a similar pixel sooner. Hence ant can drop its pixel sooner and execution time drops (Fig. 5). On the other hand Fig. 4 shows that increasing α has little effect on the quality. Therefore it is recommended to use bigger α unless the quality of the image is highly important. Also Fig. 5 shows that the more different colors are in the image, the more the execution

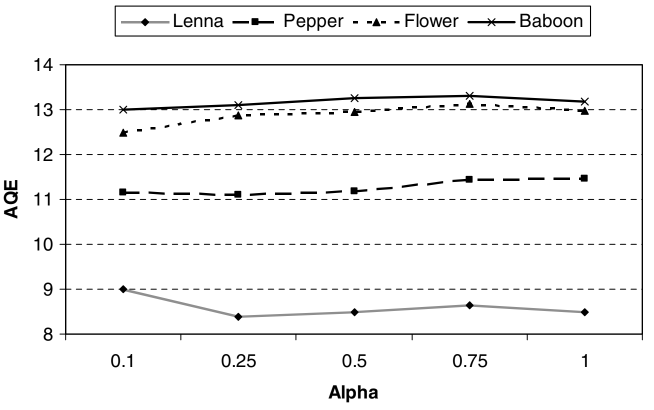


Fig. 4. Effect of α on AQE (MTh = 35, DTh = 0.6).

[1] http://sipi.usc.edu/services/database/Database.html.
[2] Average quantization error.

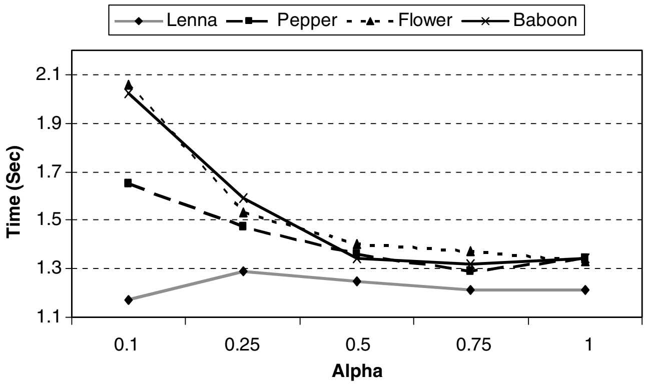


Fig. 5. Effect of α on execution time (MTh = 35, DTh = 0.6).

time will be. In other words, if the image has smooth and limited color range, the ant is able to find similar items and form the clusters sooner which leads to faster execution.

3.2.2. Drop threshold

Drop threshold, DTh, determines whether a point is a proper point for dropping or not (Fig. 2). As it was demonstrated for α, different values between 0.1 and 1.1 was tested for DTh which is a meaningful range. Results showed that ant must passes more steps due to increment of DTh because it should find more similar pixel each time which gets eventually harder. This leads to longer execution. On the other hand, increasing the DTh does not have much effect on result's quality and AQE remains almost constant.

3.2.3. Memory threshold

Memory threshold, MTh, determines whether the picked up pixel is similar enough to its closest item in the memory or not (Section 2.4.1). Figs. 6 and 7 show MTh effect on the mean of AQE and execution time for 5 runs. Other parameters are set to $\alpha = 1$ and DTh = 0.6. In most images AQE is reduced until MTh = 35 and image quality improves. Just in Lenna image, the quality improves until MTh = 20 then it deteriorate gradually. That is because the color spectrum is limited in Lenna image and lower thresholds result in least distortion. In other cases, due to increasing MTh, more pixels can be dropped by using items in ant's memory which leads to more memory updates. The more the items in memory be updated, the better the cluster representative will be and finally the AQE will be reduced. Also dropping pixels by remembering items in the memory reduces the execution time. On the other hand by increasing MTh more than 35, pixels which should form separate memory items are mixed with the previous items in memory so AQE increases. Thus MTh = 35 results in better quality of images. The quantized images resulted from MTh = 35, are shown in Fig. 12.

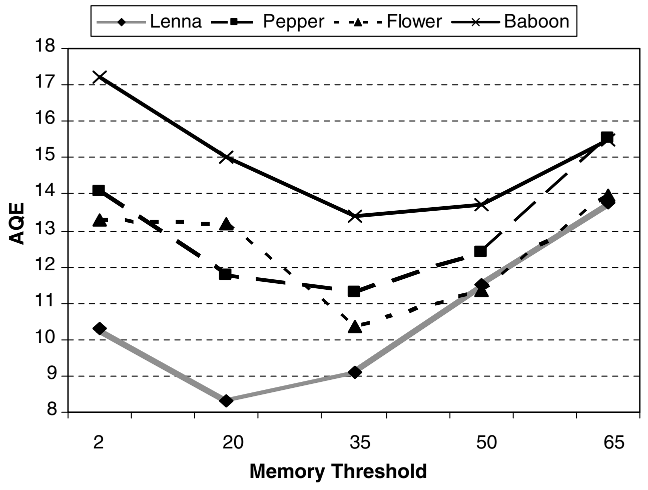


Fig. 6. MTh effect on AQE ($\alpha = 1$, DTh = 0.6).

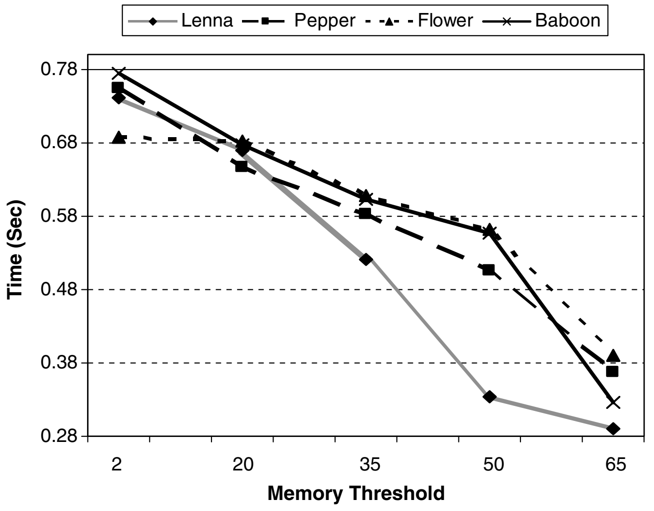


Fig. 7. MTh effect on execution time in seconds ($\alpha = 1$, DTh = 0.6).

3.2.4. Parameter K and number of iterations

Parameter K defines the number of steps the ant is allowed to pass to drop the pixel (Section 3.3). K is defined related to image dimension. For an $M \times N$ image:

$$K = \frac{(M + N) \times X}{100} \tag{6}$$

We examined X in the range of 5–50; it does not have much influence on the results. Just increasing X results in slightly increase in the execution time. Most results were obtained with acceptable execution time and quality with $X = 20$, so it is used in all runs. Because the other parameters are well-defined, the ant normally drops the pixel before reaching the K limit, so K has very little effect on the results.

The number of sampled pixels taking part in clustering, # iteration, is also related to the image dimension:

$$\# \text{ iteration} = \frac{(M \times N) \times Y}{100} \tag{7}$$

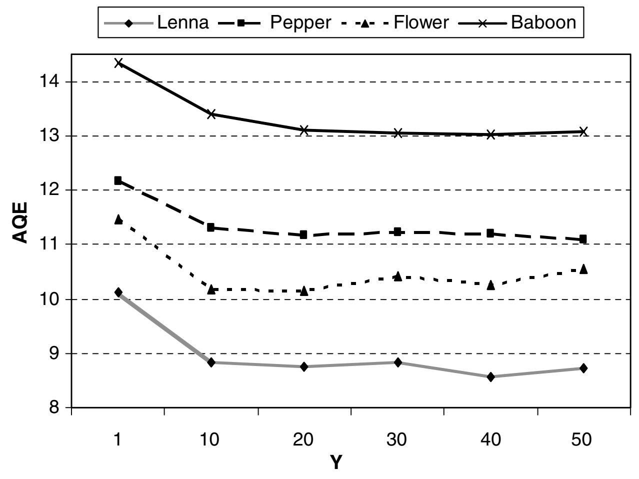


Fig. 8. Effect of increasing the percentage of modified pixels on AQE ($\alpha = 1$, DTh = 0.6, MTh = 35).

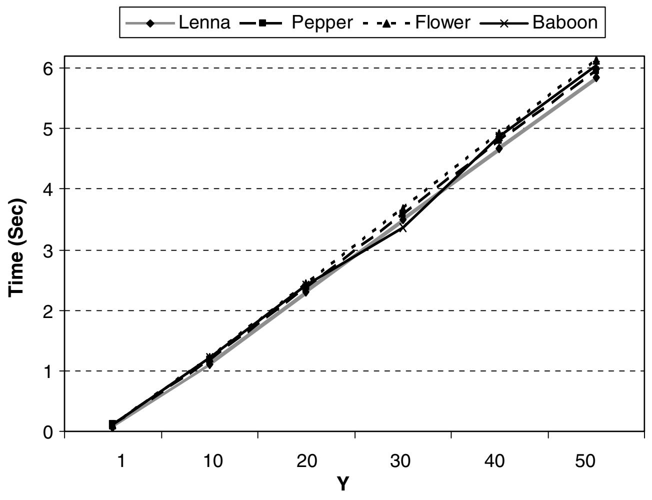


Fig. 9. Effect of increasing the percentage of modified pixels on execution time ($\alpha = 1$, DTh = 0.6, MTh = 35).

In other words, in each run, Y percent of image pixels are modified and take part in clustering process. Figs. 8 and 9 demonstrate the mean result of 5 runs. For all previous reported results, Y was considered to be 5 (Sections 3.2.1, 3.2.2, 3.2.3). As Y increases, the execution time increases too because more pixels are processed, but AQE decreases at first then it remains almost the same. By increasing the sample size, the diversity of items considered by ant is increased. This leads to improving the results at first. But around $Y = 10$ the general structure of clusters has been formed in ant's memory and visiting new pixels just improves the previous clusters slightly. After $Y = 20$ these updates does not have much effects on memory items quality and quality of results remains almost still. Thus it is recommended to set Y bigger than 10. But what value is exactly appropriate is due to the user priorities. Simply if

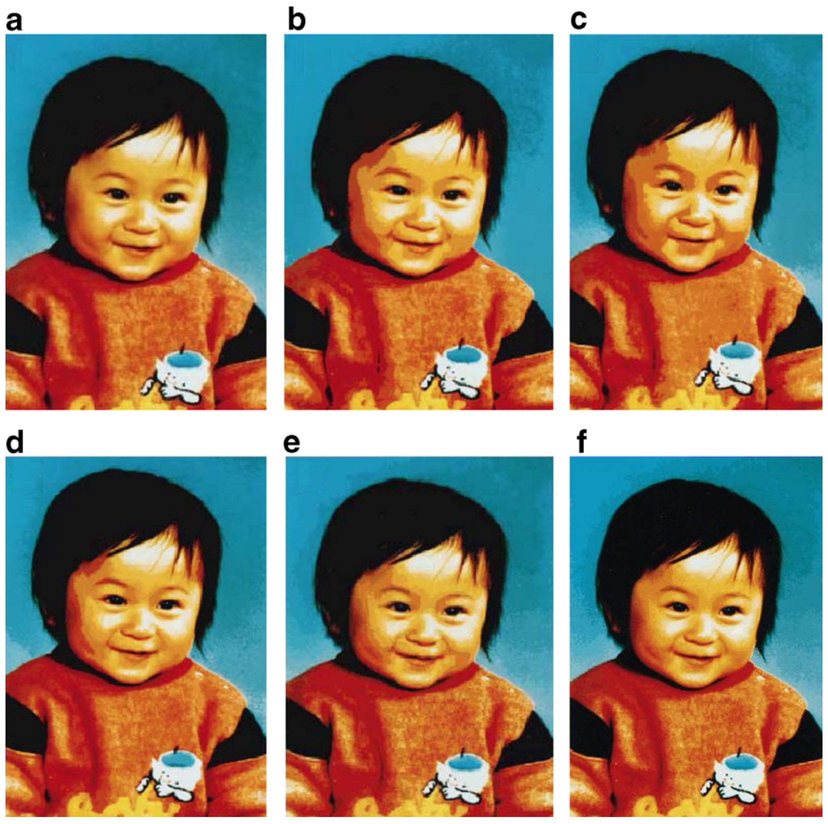


Fig. 10. Comparison of color quantization methods (a) original image, (b) Octree (Bing et al., 2004), (c) center-cut (Bing et al., 2004), (d) C-means (Bing et al., 2004), (e) Bing (Bing et al., 2004) and (f) our method.

the execution time is vital for the user, lower values of Y are recommended, but if the user cares more about the quality, Y should be set to higher values. It shows the flexibility of the algorithm.

3.3. Comparison with other techniques

In this section results of the proposed algorithm are compared with four other color reduction methods. Fig. 10 shows the results of color reduction with Octree, Center-cut, C-means and Bing's methods and our method; colors are reduced to 64 colors. Also Table 2 lists AQE of these results for more precise comparison. All results reported here for other methods are from Bing et al. (2004). As it can be seen in Fig. 10 resulting images from Octree, center-cut and C-means methods show obvious degradations. Indeed their higher AQE in Table 2 indicate this in more details. This is mainly because these methods just use color information of image in clustering. In Bing's method, first a cluster-based algorithm is used to find N initial colors. Then the color with the most pixels is used as a base color. In the next step, the rest of the colors are sorted based on a weighted products of the distance to the base color and number of pixels are calculated. First $K-1$ colors plus the base color are selected to form the palette and the remaining colors are merged with their nearest color to reinforce the final palette. So in Bing's method neighborhood information of each pixel is not considered in forming the final palette. So the quality of the results is lower than the proposed method. It can be seen through its AQE in Table 2 and some minor degradation in resulting image in Fig. 10. As it can be seen, the result of the proposed method has the least distortion and best quality among five methods. For this result parameters were set as $\alpha = 1$, DTh = 0.6, MTh = 35, $X = 20$, $Y = 10$. Also the results on the test images with the same parameter setting are included in Fig. 12.

Table 2
AQE comparison of five methods

Method	Proposed method	Bing's method	C-means	Center-cut	Octree
AQE	10.52	13.36	13.78	12.08	14.26

4. Conclusion

In this paper ant-based clustering ability for color reduction was demonstrated and a method for color reduction based on ant colony was proposed. Parameters in this method can be simply set to consider user priorities. This increases flexibility of the algorithm and also makes it easy to be used by nonprofessional users. Also with little changes, like just replacing the dissimilarity function, this solution can be extended to reduce colors in gray scale images or videos. This again shows flexibility of the proposed method.

Basically using color information together with pixels' neighborhood information in the proposed method is the main cause of improving results. Neighborhood information is deployed in three ways in our method: (1) In neighborhood function. (2) As ant movement for finding a proper place to drop the carried pixel. (3) Limiting maximum number of steps the ant can pass to drop a pixel. It was demonstrated that how using neighborhood information leads to better result comparing to other methods like

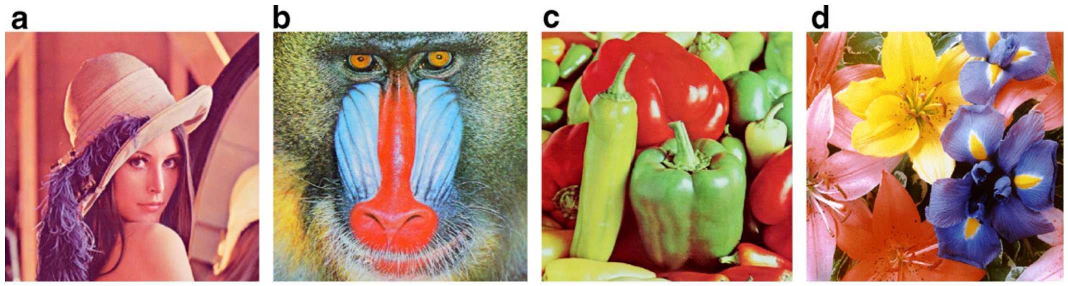


Fig. 11. Original images.

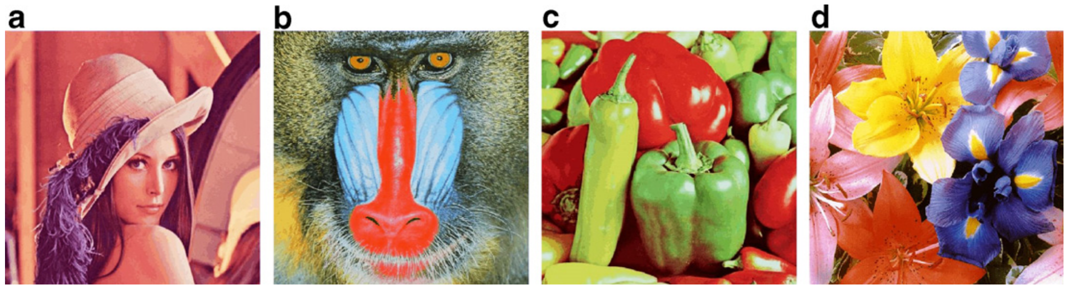


Fig. 12. Result images reduced to 64 colors by the proposed algorithm ($\alpha = 1$, DTh = 0.6, MTh = 35, $X = 20$, $Y = 10$).

C-means or center-cut which just consider color information of image.

In the proposed method mapping the image colors to resulting palette include replacing the closest item in the palette. Using more powerful mapping methods, for example dithering techniques, visual distortion can be improved. Since ants make heaps or clusters on the image itself, place of heaps can also be used to improve the visual quality of the results and a combination of ant-based clustering and a dithering technique can lead to lower degradation.

Since all solutions based on behavior of ants are inherently parallel, more research can be done on parallelizing the proposed methods. Moreover, as it has been shown in data clustering, for example (Monmarché, 1999) and (Kanade and Hall, 2003), combining ant-based clustering with other methods like fuzzy methods or C-means improves the execution time.

References

Bing, Z., Junyi, S., Qinke, P., 2004. An adjustable algorithm for color quantization. Pattern Recognition Lett. 25, 1787–1797.

Dekker, A.H., 1994. Kohonen neural networks for optimal color quantization. Network: Comput. Neural Syst. 5, 351–367.

Deneubourg, J.L., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C., Chretien, L., 1990. The dynamics of collective sorting robot-like ants and ant-like robots. In: Meyer, J.A., Wilson, S.W. (Eds.), Proc. First Conf. on Simulation of Adaptive Behavior. MIT Press/Bradford Books, pp. 356–363.

Dorigo, M., Maniezzo, V., Colorni, A., 1996. The ant system optimization by a colony of cooperating agents. IEEE Trans. Systems, Man, Cybernetics, Part B 26 (1), 29–41.

Ghanbarian, A.T., Kabir, E., 2006. An ant-based approach to color reduction. In: Fifth Workshop on Ant Colony Optimization and Swarm Intelligence, (ANTS, 2006), ANTS, Lecture Notes of Computer Science, vol. 4150, pp. 364–371.

Handl, J., 2003. Ant-based methods for tasks of clustering and topographic mapping: extensions, analysis and comparison with alternative methods. Master Thesis, University of Erlangen-Nuremberg, Germany, <http://www.handl.julia.de>.

Handl, J., Meyer, B., 2002. Improved ant-based clustering and sorting in a document retrieval interface. In: Proc. Seventh Int. Conf. on Parallel Problem Solving from Nature (PPSN VII). Lecture Notes of Computer Science, vol. 2439. Springer-Verlag, Berlin, Germany, pp. 913–923.

Handl, J., Knowles, J., Dorigo, M., 2003. Ant-based clustering: a comparative study of its relative performance with respect to k-means, average link and 1D-SOM. Technical Report TR/IRIDIA/2003-24, Universite Libre de Bruxelles.

Handl, J., Knowles, J., Dorigo, M., 2003. On the performance of ant-based clustering. Design and Application of Hybrid Intelligent Systems. In: Frontiers in Artificial Intelligence and Applications, vol. 104. IOS Press, Amsterdam, The Netherlands, pp. 204–213.

Kanade, P.M., Hall, L.O., 2003. Fuzzy ants as a clustering concept. In: 22nd Int. Conf. on North American Fuzzy Information Processing Society NAFIPS, pp 227–232.

Kuntz, P., Snyers, D., 1994. Emergent colonization and graph partitioning. In: Proc. Third Int. Conf. on Simulation of Adaptive Behavior: From Animals to Animats 3. MIT Press, Cambridge, MA, pp. 494–500.

Kuntz, P., Snyers, D., 1999. New results on an ant-based heuristic for highlighting the organization of large graphs. In: Proc. 1999 Congress on Evolutionary Computation. IEEE Press, Piscataway, NJ, pp. 1451–1458.

Kuntz, P., Snyers, D., Layzell, P., 1998. A stochastic heuristic for visualising graph clusters in a bidimensional space prior to partitioning. J. Heuristics 5 (3), 327–351.

Labroche, N., Monmarché, N., Venturini, G. Visual clustering with artificial ants colonies, In: Seventh Int. Conf. on Knowledge-Based Intelligent Information & Engineering Systems (KES 2003), pp. 332–338.

Lumer, E.D., Faieta, B., 1994. Diversity and adaptation in populations of clustering ants. In: Cliff, D., Husbands, P., Meyer, J.A., Wilson, S.W. (Eds.), Proc. Third Int. Conf. on Simulation of Adaptive Behavior: From Animals to Animats 3 (SAB 94). MIT Press, pp. 501–508.

Monmarché, N., 1999. On data clustering with artificial ants. In: AAAI-99 & GECCO-99, Workshop on Data Mining with Evolutionary Algorithms: Research Directions. Orlando, FL, pp. 23–26.

Papamarkos, N., Atsalakis, A.E., Strouthopoulos, C.P., 2002. Adaptive color reduction. IEEE Trans. Systems, MAN, Cybernetics, Part B 32 (1), 44–56.

Scheunders, P., 1997. A comparison of clustering algorithms for color image quantization. Pattern Recognition Lett., 1379–1384.

Steinberg, D., Venturini, G., Monmarché, N., Slimane, M., 1998. Discovery of clusters in numeric data by an hybridization of an ant colony with the minimum distance classification, from ant colonies to artificial ants. In: First Int. Workshop on Ant Colony Optimization (Ants 98), Bruxelles, October, 1998.

Further reading

Azzag, H., Monmarché, N., Slimane, M., Venturini, G., Guinot, C., 2003. AntTree: a new model for clustering with artificial ants. In: IEEE Congress on Evolutionary Computation, vol. 1, Canberra, Australia, pp. 2642–2647.

Hoe, K., Lai, W., Tai, T., 2002. Homogenous ants for web document similarity modeling and categorization. In: Proc. Third Int. Workshop on Ant Algorithms (ANTS 2002). Lecture Notes of Computer Science, vol. 2463. Springer-Verlag, Berlin, Germany, pp. 256–261.