ISSN 2278-3091
Volume 9, No.1.5, 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse1991.52020.pdf
https://doi.org/10.30534/ijatcse/2020/1991.52020

A Novel Content Based Image Retrieval Based on a New Approach of Color String Coding and Meta-Heuristic Algorithms

Naoufal Machhour[1], M'barek Nasri[2]
[1]University of Mohammed Premier, ESTO, Oujda, n.machhour@ump.ac.ma
[2]University of Mohammed Premier, ESTO, Oujda, nasrihome@gmail.com

ABSTRACT

Content based image retrieval (CBIR) is ranked among the most important domain of image processing. This technique allows us to find the relevant images from a query image in a very large image database. This approach is based on the visual features of the image. Therefore, the choice of these features is a very decisive factor for improving the robustness and efficiency of the CBIR system. This paper aims at the development of a novel technique which is based on the color string coding method implemented in the previous work. This new technique combines the two color spaces: RGB and HSV, and introduces the texture features to improve the results obtained previously by increasing the number of relevant images and decreasing the computational complexity and the response time whatever the size of the images. Accordingly, the image retrieval is performed using the meta-heuristic algorithms. Meanwhile, our system is evaluated based on the precision and recall measures. The obtained results show the efficiency and the performance of the proposed method compared to other CBIR systems.

Key words: Genetic algorithm, color string coding, content based image retrieval, simulated annealing.

1. INTRODUCTION

The image retrieval is an image processing technique that can be used in many fields of scientific research, such as medicine, military, industry, etc... This image processing domain becomes very important and necessary, especially with the expansion of image databases due to the new technologies and the wide use of the internet.

The first technique was based on keywords or tags which is a very expensive method, because it requires a laborious manual work and time-consuming. Quickly, it appears a new technique that is based on the visual features of the image. It is the most used and the efficient method so far. Therefore, content based image retrieval (CBIR) is a technique which makes it possible to find similar images to a query image (QI). This approach uses the color, texture or shape descriptors [1], [2], [3].

Therefore, a CBIR system is a system that is able to retrieve the most relevant images that are similar to a query image in a large image database [4]. In general, the CBIR systems follow two important steps. The first one which is an offline step consists in the extraction of the visual features of all images in the image base and storing them in the features database. The second step, which is an online process, is performed by extracting the query image features, then calculating the similarity measures between the descriptors of the database images and those of the query image to get the relevant images. Thus, the performance of the CBIR systems depends on the features which are extracted, how are used and which retrieval algorithm implemented. Figure 1 illustrates a diagram of the CBIR system.

2. RELATED WORKS

Many works have been done in this area to improve the performance of these CBIR systems. As already mentioned, all these works use several visual features of the image at the same time. However, merging multiple features gives good results, but the response time is high and the calculations are too expensive. In recent research, Balaji *et al*. [5] created a CBIR system based on tag clustering. They use the tags which are associated with the images to group these latters then extracted the image features such as color, texture and shape. Therefore, the image retrieval is performed based on the membership value in the tag clusters. Rao *et al*. [6] modified the color histogram to get the spatial layout information of colors and introduced three histograms: angular, annular and hybrid color histograms. Pass *et al*. [7] used a technique to compare images based on the histogram refinement by comparing only the pixels in the same classes. Liu *et al*. [8] described a new method based on color difference histogram; they defined new descriptor combining edge orientation, color and perceptually uniform color difference.

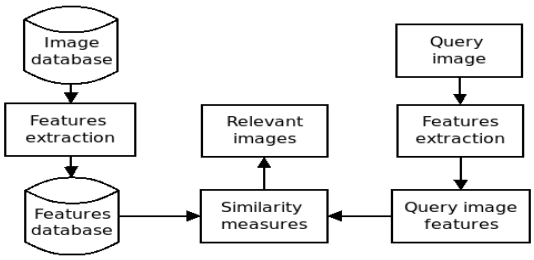


Figure 1: Block diagram of the CBIR system

Singha *et al*. [9] combined texture and color features that are extracted from color histogram and wavelet transformation. Also, Nazir *et al*. [10] extracted color and texture features using the edge histogram descriptor and discrete wavelet transform. Yue *et al*. [2] defined a quantification of the HSV color space and created feature vectors using the co-occurrence matrix.

Ashraf *et al*. [11] used the bandelet transform to represent the image features and applied the artificial neural network for image retrieval. Alsmadi [12] fused color, texture and shape features; then the similarity measure is calculated using the genetic algorithm with iterated local search. According to Kushwaha *et al*. [13], the appropriate and adequate features for image retrieval are selected using the genetic algorithm.

Also, and in related domain, Jha *et al.* [14] used the CBIR techniques for historical monuments retrieval. The purpose of their work is to create an efficient system based on the machine learning techniques to retrieve the correct monuments images. They commence by extracting the image features which are the color features and Histogram of Oriented Gradient texture feature. Then, they performed an unsupervised machine learning clustering to avoid the search for matching similarity in the whole image dataset.

Lin *et al*. [15] applied the color string comparison after converting the video frames to a color string. In Jenni *et al*. [16], all the images in the database are classified into different groups using support vector machine, then the image retrieval is done based on a simple similarity measure between strings representing images. Machhour *et al.* [17] modified the criteria of the color string coding to increase the difference between images by arranging the colors in seven color series and applied the genetic algorithm for image retrieval.

3. PROPOSED METHOD

In the work in hand, we implemented a new color coding technique to represent the image features, then the meta-heuristic algorithms for image retrieval. This new approach which is based on the color string coding method will increase the number of relevant images and improve the response and execution time of the previous methods [15]. The proposed CBIR system is presented in Figure 2.

In this method, we begin by extracting the color features, performing a segmentation process using the HSV color space and calculating the texture features using the gray level co-occurrence matrix (GLCM), then applying the new color string coding technique to represent the images. Finally, we will apply the genetic algorithm and the simulated annealing to find the relevant images. The comparison of the obtained results using the two algorithms will be done based on the precision-recall measurements and retrieval time.

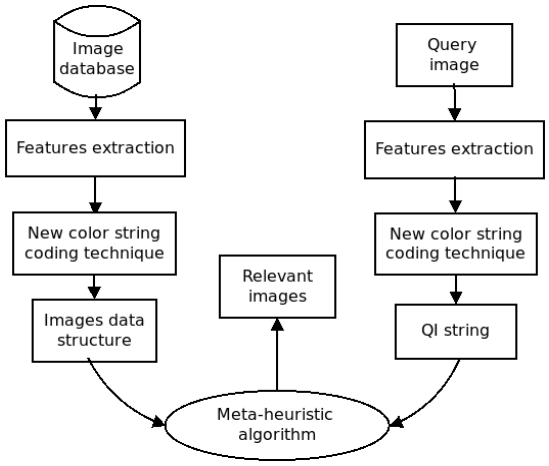


Figure 2: Diagram of the proposed CBIR system

3.1 Color Features Extraction

The color features are widely used in image retrieval. They are very efficient and give good results, especially if they are merged with other descriptors. As mentioned in the previous work, the first step of color features extraction is to normalize the size of all images (e.g. 64×64). For this, we use the bilinear interpolation technique [18] by adding pixels using the average of the intensity values of surrounding pixels. This technique gives a smoother result than the nearest neighbor method [19]. Also, it produces medium-quality results, but it is very fast than the bi-cubic interpolation. Next, we create three matrices (*R, G* and *B*) for each image. Each one contains the values of one color: red, green or blue. The goal of this separation is to facilitate comparing the three intensities for each pixel.

3.2 Color String Coding

After creating these three matrices, we compare them element by element to determine the color value that will be taken to create the final matrix which contains the strongest intensities. The previous method [15], [16], determines six criteria for coding colors. This method does not specify some exceptional cases, and it can assign two characters for the same color series. For example, the blue color series can be represented by two characters *"B"* and *"C"* [16]. In our previous work [17], we modified some criteria, and added another one to describe all the possible cases in order to increase the difference between the strings representing the

images in the database. Our codification is based on the seven criteria below [17]:

1) *if max(R, G, B) = R, take the character "R";*
2) *if max(R, G, B) = G, take the character "G"*
3) *if max(R, G, B) = B, take the character "B"*
4) *if max(R, G, B) = R and G, take the character "A"*
5) *if max(R, G, B) = R and B, take the character "C"*
6) *if max(R, G, B) = B and G, take the character "D"*
7) *if R = G = B, take the character "E"*

Where, (*R), (G)* and (*B)* are the values of the red, green and blue colors.

In this codification, we can see that the blue color series are represented only by one character: *"B"*. Now, we get a single matrix which contains only the characters *"R", "G", "B", "A", "C", "D"* and *"E"*. Then, we proceed by concatenating all the matrix elements to obtain finally a single string representing the image (e.g. "RRRRRCCCCC...BBBGGG").

3.3 New Approach of the Color String Coding Method

As aforementioned, each image is represented by a single string. Therefore, the strings become longer with larger images. This method gives good results with a relatively average execution time if the size of the images is small. The execution time becomes very high if we want to apply it on large images. Consequently, the first improvement that we brought to this method is reducing the response time, whatever the size of the images, by reducing the length of strings.

The new approach adopted in this work consists of calculating the occurrences of each character representing a color series in the string. This occurrence value becomes a coefficient for each color series. So, whatever the size of the image, we get a string of the following form (1):

$$"\alpha_1 R\alpha_2 G\alpha_3 B\alpha_4 A\alpha_5 C\alpha_6 D\alpha_7 E" \quad (1)$$

Where, (α_i) represents the occurrence values of each color series.

For example, an image of size 16×16 becomes a string of 256 characters, then another string like the following form: *"80R40G20B50A40C10D16E"*. In this representation, we obtain a string containing a maximum of 21 characters instead of 256. Consequently, we conclude that the string length with this approach is much smaller than that with the previous coding method [17]. Also, it is clear that the image size will not influence too much the final string length. However, we can say that this approach brought a considerable improvement to the execution time.

Therefore, as a second improvement to our previous method, we will introduce other features to increase the number of relevant images. For this, we utilize the texture features and a simple segmentation technique which is based on the HSV color space. We choose these two descriptors because they are extracted directly from the RGB color space based on easy and simple calculation techniques.

A. Segmentation Process

Proposed by Smith in 1978 [20] and designed by computer graphics researchers, HSV (hue, saturation and value) is an alternative representation of the RGB color space. The HSV color space can represent the human color perception and it is very close to our visual system. In fact, each image contains different objects which have distinct colors (hues) and luminosities. Therefore, we can use these features to separate different image areas. Also, these features which are the hue and the luminosity are expressed as a linear combination of the R, G and B channels. In HSV space, the hue and luminosity correspond to the two channels which are the Hue and Value channels. In this work, we use a simple, quick and efficient segmentation method based on a mere thresholding applied to the HSV channels. In this segmentation process, we begin by converting the RGB color space to the HSV one. The easiest method is to use the formulas developed by Su *et al.* [21] as in (2), (3) and (4):

$$H = \cos^{-1}\frac{1}{2}\left[(R-G)+(R-B)\right] \Big/ \sqrt{(R-G)^2-(G-B)(R-B)} \quad (2)$$

$$S = 1-(3[\min(R,G,B)]/(R+G+B)) \quad (3)$$

$$V = (R+G+B)/3 \quad (4)$$

Where (*R*), (*G)* and (*B)* represent the red, green and blue color values respectively; also, (*H*), *(S)* and (*V)* are the values of the hue, saturation and value components respectively. Then, we apply a threshold on the Hue and Value channels to separate the objects from the background. We get finally two matrices containing the new values of the Hue and Value components. The threshold value of the Hue and Value channels is chosen in this work as the mean of these features respectively. The comparison between these values allows us to select the adequate element for the next coding step. The comparison criteria are presented below:

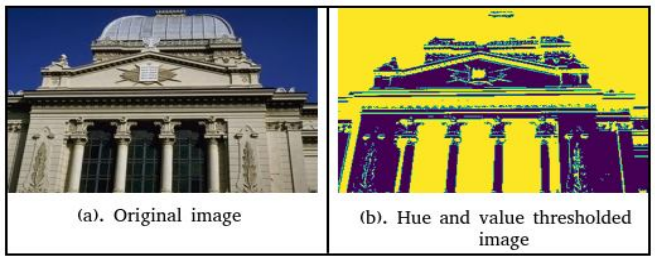
1) *if $H > T_h$ and $V < T_v$, take the character "H"*
2) *if $H < T_h$ and $V < T_v$, take the character "V"*
3) *if $H > T_h$ and $V > T_v$, take the character "F"*
4) *if $H < T_h$ and $V > T_v$, take the character "I"*

Where (T_h) and (T_v) are the threshold values of the hue (*H*) and value (*V*) components respectively.

Figure 3 illustrate the original image and the hue and value thresholded image. Next, we calculate the occurrences of each character (*"H", "V", "F"* and *"I"*) to constitute a new string for this segmentation phase as shown in (5) below:

$$"\alpha_8 H \alpha_9 V \alpha_{10} F \alpha_{11} I" \quad (5)$$

Where, α_8, α_9, α_{10} and α_{11} are the occurrence values of the characters representing the thresholded image.



(a). Original image | (b). Hue and value thresholded image

Figure 3: (a). Original (b). Hue and value thresholded image

B. Texture Features Extraction

Texture is also an important low level features. This descriptor gives more details about specific regions in image, especially the arrangements of colors; they interpret a homogeneous aspect of the surface of an object in the image [22]. In this work we use the grey level co-occurrence matrix (GLCM) to extract the texture features [23]. Equations (6), (7), (8) and (9) are extracted from the GLCM to describe the texture:

$$Energy = \sum_{i,j=0}^{N-1} p_{ij} \quad (6)$$

$$Contrast = \sum_{i,j=0}^{N-1} p_{ij}(i-j)^2 \quad (7)$$

$$Entropy = \sum_{i,j=0}^{N-1} p_{ij} \log(p_{ij}) \quad (8)$$

$$Homogeneity = \sum_{i,j=0}^{N-1} p_{ij} / (1+|i-j|) \quad (9)$$

Where, (N) is the number of grey levels and (p_{ij}) is the $(i,j)^{th}$ entry in a grey level.

Next, we assign the characters *"J"*, *"K"*, *"L"* and *"M"* to the values of the energy, contrast, entropy and homogeneity respectively. Then, we get another string as in (10) below:

$$"\alpha_{12} J \alpha_{13} K \alpha_{14} L \alpha_{15} M" \quad (10)$$

Where, α_{12}, α_{13}, α_{14} and α_{15} are the values of the energy, contrast, entropy and homogeneity respectively.

The coefficient values of color and texture features are not in the same interval, which will influence the retrieval results. To overcome this problem, we apply a technique to the texture coefficients to bring them to the same color range.

This interval will be between 0 and *N* where *N* is the size of the image. The new coefficient value (α'_i) of the texture feature is calculated from the old one (α_i) and presented in (11) below:

$$\alpha'_i = \alpha_i * \left(\sum_{j=1}^{7} \alpha_j \Big/ \sum_{k=12}^{15} \alpha_k\right) \quad (11)$$

Where, α_j and α_k are the color and texture coefficients respectively.

C. New color string coding formula

After extracting the color and texture features, performing the segmentation process and creating the strings which code each descriptor, we concatenate the three strings presented in (1), (5) and (10) to get the new string coding formula of the image as shown in (12):

$$"\alpha_1 R \alpha_2 G \alpha_3 B \alpha_4 A \alpha_5 C \alpha_6 D \alpha_7 E \alpha_8 H \alpha_9 V \alpha_{10} F \alpha_{11} I \alpha J_{12} \alpha_{13} K \alpha_{14} L \alpha_{15} M" \quad (12)$$

3.4 Proposed Algorithm

Classical and common systems use a simple comparison method to find the results. As we work in a large population, meta-heuristic methods are very efficient in this type of problem. In our previous work, we have implemented the genetic algorithm which gives good results. In this paper, we also apply the genetic algorithm and the simulated annealing for image retrieval.

4. PROPOSED META-HEURISTIC ALGORITHM

Over the last years, meta-heuristic algorithms have been known as a very efficient and powerful tool in the field of research and optimization. Therefore, from a randomly generated population, these algorithms exploit the competition between the population components to find the optimal solution. These algorithms include genetic algorithm (GA), ant colony optimization (ACO), particle swarm optimization (PSO), gravitational search algorithm (GSA), differential evolutionary (DE), simulated annealing (SA), etc... In this work, we use the simulated annealing and genetic algorithm as a meta-heuristic approach.

4.1 Simulated Annealing Algorithm

The simulated annealing algorithm (SA) is developed by Kirkpatrick *et al.* in 1983 [24] which is an optimization method based on a physical phenomenon (the annealing of solids). Heating a solid and cooling it slowly are the two steps of its mechanism. Therefore, the simulated annealing algorithm provides good results with a lower execution time and low number of iterations. Furthermore, it gives a global optimum and avoids getting trapped in a local minimum as

mentioned in Dekkers *et al.* [25]. The pseudo code of the simulated annealing algorithm is presented in Figure 4. The goal of the simulated annealing is to find a state in the solution space which minimizes the objective function.

4.2 Objective Function

In the work in hand, we have created our system using a new technique which codes each image with a string as described in (12). The objective function developed in (13) and used in this algorithm is the Euclidean distance between the database images and the query image. Consequently, the solution which minimizes this objective function contains all the information about the relevant image.

$$f = \sqrt{\sum_{i=1}^{15} (\alpha_i - \alpha_{qi})^2} \tag{13}$$

Where, α_i and α_{qi} are respectively the coefficient values of the characters representing the database images and the query image in (12).

Select randomly an initial state $x \in S$
Initialization : $T \leftarrow T_i$
Define a value of T_f
while *not termination criteria* **do**
  Set the iteration counter $n \leftarrow 0$
  while $n < N_T$ **do**
    Generate x' *a neighbour of* x
    Calculate $\delta E \leftarrow f(x') - f(x)$
    if $\delta E < 0$ **then**
      $x \leftarrow x'$
    else
      $r \leftarrow$ *random value between* 0 *and* 1
      if $exp(-\delta E/T) > r$ **then**
        $x \leftarrow x'$
      end if
    end if
    $n \leftarrow n + 1$
  end while
  Set the value of ϕ
  Decrease the control parameter $T \leftarrow T * \phi$
end while

Figure 4: The simulated annealing pseudo code

4.3 Description of the Simulated Annealing Algorithm

In the proposed algorithm, the solution space (*S*), is the finite set of solutions which are the database images. Therefore, this algorithm is a kind of local search which starts with a random initial solution (*x*) which is an image in the database. Then it selects a neighbour (*x'*) of the initial solution and calculates the energy level value as in (14) which is a change in the cost function (*f*) developed in (13).

$$\delta E = f(x') - f(x) \tag{14}$$

Therefore, we replace the current solution (*x*) by its neighbour (*x'*) if there is a reduction in the energy level. Otherwise, we keep the initial solution. To avoid being trapped in the local optima, we accept to move to the neighbour solution (*x'*) despite the increase of the energy level (δE). Whether, this change in the solution is accepted or rejected based on a probability called acceptance function as presented in (15).

$$\exp(-\delta E/T) > r \tag{15}$$

Where (*r*) is a random value between 0 and 1 and (*T*) is the control parameter representing the temperature of the annealing process.

During these optimization steps, the system cooling is linked to the decreasing of the value of the control parameter (*T*) which is based on the cooling factor (φ) as shown in (16):

$$T = T * \varphi \tag{16}$$

The SA algorithm begins with an initial value (T_i) of (*T*) which is relatively high and continues trying at each temperature many neighbourhood moves (N_T) using an iteration counter (*n*) from 0 to N_T. The temperature decreases after each iteration of the external loop until reaching the final stability temperature (T_f) of (*T*) where the system converges to the optimal solution.

4.4 Genetic algorithm

The second algorithm used in this work for image retrieval is the genetic algorithm (GA). As presented in our previous work [17], the GA is a research method based on the evolutionary concept using natural operations in large population: selection, crossover, mutation and the survival individual [26]. Therefore, the pseudo code and all the steps of the GA are presented and explained in [17]. In the work in hand, our population is the image database which is composed of a set of chromosomes. Each chromosome contains a number of genes. As described above, all the images in the database are coded using the new color string coding method as in (12). Thus, in the GA, all the genes are represented by an array which contains the coefficients of all the characters in (12). Figure 5 and Figure 6 represent an example of a chromosome and a gene respectively. The fitness function of the GA is also the Euclidean distance between the database images and the query image as developed in (13).

All the parameters setting of the simulated annealing and the genetic algorithm implemented in this paper are presented in Table 1 and Table 2 respectively.

$Chrom_i$	g_1	g_2	g_3	g_4	g_5	g_6	g_7	g_8

Figure 5: Chromosome of the size 8

Where, (*g1, g2…g8*) are the genes of the chromosome.

$Gene_i$	α_1	α_2	α_3	α_4	α_5	α_6	.....	α_{15}

Figure 6: Gene form

Where, (α_1, α_2 ... α_{15}) are the coefficients of the string representing the image.

Table 1: Parameters setting of the SA

Parameter	Value
Initial acceptance probability	0.7
Final acceptance probability	0.001
Initial temperature (T_i)	-1/log(0.7)
Final temperature (T_f)	-1/log(0.001)
Number of cycles (termination criteria)	100
Number of trials per cycle (N_T)	100
Cooling factor (φ)	0.97
Control parameter (r)	Between 0 and 1
Solution space size (S)	1000
Number of solutions	25

Table 2: Parameters setting of the GA

Parameter	Value
Population size	1000
Chromosome size	25
Termination criteria (number of generation)	1000
Crossover rate	0.75
Mutation rate	0.01

5. EXPERIMENTAL EVALUATION AND RESULTS

5.1 Data sets and experiment

For the experiment, we choose an image dataset which contains a set of varied images. For this, the Corel image dataset is the adequate one to evaluate the proposed CBIR system. This database contains 1000 images divided into 10 classes; each of them contains 100 images of the size 384 × 256 or 256 × 384. These classes are: Africa, buses, mountains, horses, buildings, food, elephants, flowers, dinosaurs, and beach. Figure 7 presents sample images of the Corel dataset. These image classes belong to different field which give us the possibility to evaluate the performance of our system using different type of images. Also, this image dataset is widely used in related works. On the one hand it allows us to test the effectiveness of our method; on the other hand it gives us the possibility to compare our results with those of the previous works. Therefore, the performance of our system is tested based on a set of query images chosen randomly from the Corel image dataset. For this, we begin by extracting the features of the dataset images and storing them in a database while respecting an appropriate data structure.

We used color signature to code the images based on the Color String Coding method and to apply a segmentation technique by converting them into the HSV color space. We applied a simple and efficient thresholding technique to the Hue and Value channels to extract all the objects in the image. Also, the texture features extraction is done using the color signature by calculating the GLCM. This latter allowed us to calculate the 4 values of the texture which are: energy, contrast, entropy and homogeneity.

Figure 7: Sample images of Corel dataset

After performing this features extraction step, we obtained 15 components, 7 for color, 4 for segmentation and 4 for texture. Consequently, our new coding method is developed by assigning a unique character to each component as shown in (12). Thus, the values of each feature became a coefficient which will be used to calculate the distance between the images.

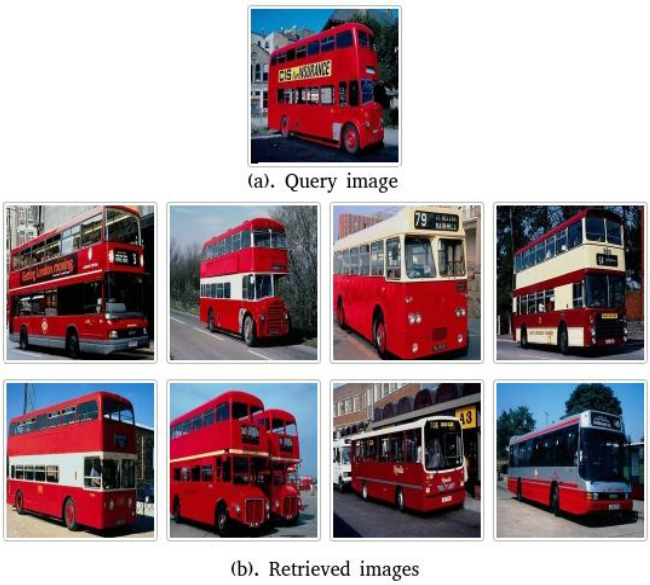
For the image retrieval, we choose an image from the data base; we extract its features using the same technique explained above. Then the SA chooses randomly an image from the dataset to start the search, and tries to find the neighbours which have the minimum distance from the query image based on the decreasing in the cost of the objective function. This search process is repeated as the temperature decreases until reaching the termination criterion. We repeat this algorithm until getting the desired number of similar images to the query image.

In the second retrieval method implemented in this work, the GA generates a random population from the image base, and then composing the chromosomes containing a set of genes which represents the number of desired images. The number of chromosomes is the ratio of the size of the population and the number of images desired. Therefore, these chromosomes are subjected to the natural operations: selection, crossover and mutation which are repeated until reaching the maximum number of iterations to provide at the end the fittest chromosome which contains the information of the relevant

images. Figure 8 shows some retrieved images from the bus query image.

5.2 Results and evaluation

In an information retrieval system, especially the image search system, we are interested in relevant results. So, to evaluate such a system, we look for the precision of the answer which



(a). Query image

(b). Retrieved images

Figure 8: (a). Bus query image (b). Some retrieved images

means the evaluation of research performance. Thus, to evaluate the effectiveness of our CBIR system, we calculate two measures: precision and recall. These measures are the most used methods to evaluate the accuracy of retrieved image.

A. Precision

Used to determine the accuracy of image retrieval, precision (Pr) as developed in (17) is the ratio of the relevant results in all images found by the number of images retrieved.

$$\Pr = \frac{NumberOfrelevant\ \mathrm{Im}ages\ \mathrm{Re}trieved}{TotalNumberOf\ \mathrm{Im}ages\ \mathrm{Re}trieved} \qquad (17)$$

B. Recall

Recall (Re) as presented in (18) is the ratio of the number of relevant results in the set of images retrieved by the number of relevant images in the database. It is calculated to show the robustness of the image retrieval method.

$$\mathrm{Re} = \frac{NumberOfrelevant\ \mathrm{Im}ages\ \mathrm{Re}trieved}{NumberOf\ \mathrm{Re}levant\ \mathrm{Im}agesInDatabase} \qquad (18)$$

The population size is 1000 and the number of desired images is 25. Table 3 shows the average of the recall and precision values that was calculated for some query images for all dataset classes using the two algorithms: SA and GA. Figure 9 illustrate the precision measurements for our CBIR system.

Table 3: Recall and Precision measurements

	SA		GA	
Classes	**Precision**	**Recall**	**Precision**	**Recall**
Africa	0.76	0.19	0.76	0.19
Mountains	0.76	0.19	0.8	0.2
Elephants	0.8	0.21	0.84	0.22
Buildings	0.72	0.18	0.76	0.19
Horses	0.92	0.23	0.88	0.22
Buses	0.96	0.24	0.96	0.24
Beach	0.72	0.18	0.72	0.18
Dinosaurs	1	0.25	0.96	0.24
Flowers	0.92	0.22	0.96	0.23
Food	0.76	0.19	0.8	0.2

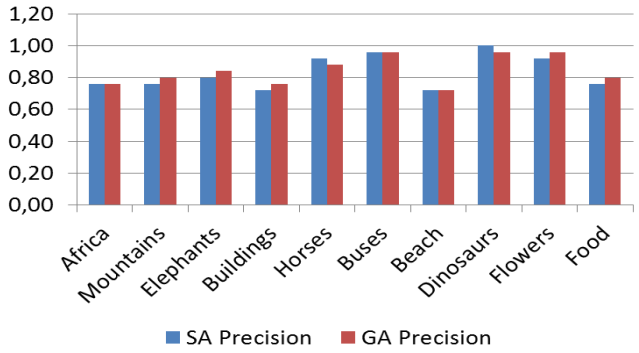


Figure 9: Precision graph of the SA and GA

C. Retrieval time

Another evaluation factor which mentioned in this work is the query time. In this study, the experiments are done using a computer with Processor: Intel® Core™ I5-6300U CPU @ 2.40 GHz x 4, 8 GiB of memory and running on Ubuntu 19.10 operating system (64-bit). The Python 3.7.5 interpreter is used for simulation and reporting the retrieval time for the two algorithms: SA and GA. Table 4 presents the results obtained in terms of precision-recall and response time.

6. DISCUSSION

In this study, we made a comparison of our CBIR system with other systems which are created based on different methods. We chose these methods for the comparison of our work, because they use the same image base which contains 10 image classes belonging to different domains. This comparison allows us to know the performance and the limits of our method for each image class.

Table 5 shows the average precision (Pr) obtained in this work compared with other CBIR systems for each class. As the results in Table 5 shows, we can clearly conclude that our method provides competitive results for some groups of

images and better for other groups. But, in general, the average precision for all classes shows the efficiency, stability and performance of our system.

Also Table 5 presents a comparison of the average recall value (Re) of our method with other systems. The results are very good compared to most other methods. The recall value which is less good compared to the work of Alsmadi [12] and Madhavi *et al*. [27] is due to the choice of the number of desired images which is 25 and not to the performance of the system. On the other hand, this choice reduces the query time and makes our system fast.

As already mentioned above, the average values of precision and recall are: 0.844 and 0.211using the genetic algorithm and 0.83 and 0.208 using the simulated annealing respectively. These results exceed most methods because these latters limit the range of the image features. On the other hand, our method provides less good results than the Alsmdi's method [11] because the author combines several features which are color histogram, 8 texture features which are extracted from the GLCM and the shape features using different techniques such as filters and edge detection methods. In this work, we used the color signature to extract a set of robust descriptors to describe the image.

Table 4: Retrieval time of the SA and GA

Meta-heuristic algorithm	**SA**	**GA**
Average precision	0.83	0.844
Average recall	0.208	0.211
CPU query time (s)	5	130

Table 5: Comparison of results between our CBIR system and other methods based on average precision and recall

	Lin *et al.* [1]		**El Alami** [28]		**Machhour *et al.*** [17]		**Ashraf et al.** [29]		**Madhavi *et al.*** [27]		**Alsmadi** [12]		**Proposed method (SA)**		**Proposed method (GA)**	
Classes	**Pr**	**Re**	**Pr**	**Re**	**Pr**	**Re**	**Pr**	**Re**	**Pr**	**Re**	**Pr**	**Re**	**Pr**	**Re**	**Pr**	**Re**
Africa	0.68	0.14	0.70	0.15	0.6	0.15	0.8	0.16	0.828	0.706	0.838	0.73	0.76	0.19	0.76	0.19
Mountains	0.52	0.21	0.53	0.22	1.0	0.24	0.7	0.14	0.811	0.732	0.82	0.75	0.76	0.19	0.8	0.2
Elephants	0.65	0.14	0.67	0.15	0.75	0.18	0.9	0.18	0.727	0.533	0.83	0.58	0.8	0.21	0.84	0.22
Buildings	0.54	0.17	0.57	0.18	0.8	0.9	0.75	0.15	0.632	0.585	0.755	0.62	0.72	0.18	0.76	0.19
Horses	0.80	0.10	0.83	0.13	0.8	0.19	0.9	0.18	0.951	0.848	0.96	0.85	0.92	0.23	0.88	0.22
Buses	0.88	0.12	0.87	0.11	1.0	0.24	0.9	0.18	0.846	0.733	0.96	0.75	0.96	0.24	0.96	0.24
Beach	0.54	0.19	0.56	0.19	0.6	0.15	0.75	0.15	0.892	0.805	0.90	0.815	0.72	0.18	0.72	0.18
Dinosaurs	0.99	0,10	0.97	0.09	0.6	0.15	1.0	0.2	0.828	0.726	0.99	0.75	1.0	0.25	0.96	0.24
Flowers	0.89	0.11	0.91	0.11	0.6	0.15	0.8	0.16	0.917	0.647	0.96	0.66	0.92	0.22	0.96	0.23
Food	0.73	0.13	0.74	0.13	0.6	0.15	0.8	0.16	0.871	0.600	0.87	0.62	0.76	0.19	0.8	0.2
Average	0.722	0.144	0.735	0.146	0.735	0.179	0.83	0.166	0.83	0.691	0.888	0.712	0.83	0.208	0.844	0.211

The texture features are extracted from the GLCM while the segmentation process was developed based on a threshold technique applied to the HSV color space. Also the new approach of the color string coding method based on colors has allowed us to improve the results obtained by the previous technique in terms of precision and recall in a fast and optimal execution time whatever the size of images.

On the other hand, we have implemented two meta-heuristic algorithms which are the SA and GA for image retrieval and optimization. These algorithms give efficient results in large populations. As shown in Table 4 which presents a comparison between SA and GA in terms of precision-recall and response time. The genetic algorithms are very powerful in research and optimization but with a longer execution time while the SA gives competitive results but with a very short execution time. Consequently, these algorithms can provide better results in much larger populations. Clearly, the combination of the new coding technique and the meta-heuristic algorithms show the efficiency of our CBIR system in terms of precision and retrieval time.

7. CONCLUSION

In this work, we have created an efficient and powerful CBIR system based on a new approach of our previous method which is the color string coding technique and the meta-heuristic algorithms for image retrieval. This study brought a big improvement to our previous method in term of precision-recall and query time. In the work in hand, we used the color signature to code the images by arranging them in seven color series, then we performed a segmentation based on a thresholding technique applied to the HSV color space, and finally, we extracted four texture features from the GLCM matrix. Meanwhile, two meta-heuristic algorithms are implemented efficiently to provide similar images to a query image. The new method of coding images which is based on two color spaces (RGB and HSV) and texture combined with meta-heuristic algorithms gave competitive results in terms of precision-recall and search time whatever the size of the images. As already demonstrated, the precision and recall values obtained in this study using the GA or SA supersede those of several methods and show the efficiency of the

proposed CBIR for image retrieval in large and varied image datasets. In the future, other features and hybrid meta-heuristic algorithms will be employed to improve the content based image retrieval system.

REFERENCES

1. C.-H. Lin, R.-T. Chen, and Y.-K. Chan, **A smart content-based image retrieval system based on color and texture feature**. *Image Vis. Comput.*, vol. 27, no. 6, pp. 658–665, 2009.
2. J. Yue, Z. Li, L. Liu, and Z. Fu, **Content-based image retrieval using color and texture fused features.** *Math. Comput. Model.*, vol. 54, no. 3–4, pp. 1121–1127, 2011, doi: 10.1016/j.mcm.2010.11.044.
3. X.-Y. Wang, H.-Y. Yang, and D.-M. Li, **A new content-based image retrieval technique using color and texture information**. *Comput. Electr. Eng.*, vol. 39, no. 3, pp. 746–761, Apr. 2013, doi: 10.1016/J.COMPELECENG.2013.01.005.
4. G. Carneiro, A. B. Chan, P. J. Moreno, and N. Vasconcelos, **Supervised learning of semantic classes for image annotation and retrieval**. *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 3, pp. 394–410, 2007, doi: 10.1109/TPAMI.2007.61.
5. B. Saravana Balaji, V. Krishna Kumar, and A. N. Ahmed, **Semantically enriched tag clustering and image feature based image retrieval system.** *Int. J. Adv. Trends Comput. Sci. Eng.*, vol. 8, no. 1, pp. 138–141, 2019, doi: 10.30534/ijatcse/2019/2381.22019.
6. A. Rao, R. K. Srihari, and Z. Zhang, **Spatial color histograms for content-based image retrieval**. in *Proceedings 11th International Conference on Tools with Artificial Intelligence*, 1999, pp. 183–186.
7. G. Pass and R. Zabih, **Histogram refinement for content-based image retrieval**. in *Proceedings Third IEEE Workshop on Applications of Computer Vision. WACV'96*, 1996, pp. 96–102.
8. G.-H. Liu and J.-Y. Yang, **Content-based image retrieval using color difference histogram**. *Pattern Recognit.*, vol. 46, no. 1, pp. 188–198, 2013.
9. M. Singha and K. Hemachandran, **Content based image retrieval using color and texture**. *Signal Image Process.*, vol. 3, no. 1, p. 39, 2012.
10. A. Nazir, R. Ashraf, T. Hamdani, and N. Ali, **Content based image retrieval system by using HSV color histogram, discrete wavelet transform and edge histogram descriptor**. in *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, 2018, pp. 1–6.
11. R. Ashraf, K. B. Bajwa, and T. Mahmood, **Content-based Image Retrieval by Exploring Bandletized Regions through Support Vector Machines**. *J. Inf. Sci. Eng.*, vol. 32, no. 2, pp. 245–269, 2016.
12. M. K. Alsmadi, **Query-sensitive similarity measure for content-based image retrieval using meta-heuristic algorithm**. *J. King Saud Univ. Inf. Sci.*, vol. 30, no. 3, pp. 373–381, 2018.
13. P. Kushwaha and R. R. **Welekar, Feature Selection for Image Retrieval based on Genetic Algorithm.** *IJIMAI*, vol. 4, no. 2, pp. 16–21, 2016.
14. J. Jha and S. S. Bhaduaria, **A novel approach for retrieval of historical monuments images using visual contents and unsupervised machine learning**. *Int. J. Adv. Trends Comput. Sci. Eng.*, vol. 9, no. 3, pp. 3563–3569, 2020, doi: 10.30534/ijatcse/2020/162932020.
15. C. Lin and C. Su, **Using Color Strings Comparison for Video Frames Retrieval**. in *2009 International Conference on Information and Multimedia Technology*, 2009, pp. 211–215, doi: 10.1109/ICIMT.2009.30.
16. K. Jenni, S. Mandala, and M. S. Sunar, **Content based image retrieval using colour strings comparison**. *Procedia Comput. Sci.*, vol. 50, pp. 374–379, 2015, doi: 10.1016/j.procs.2015.04.032.
17. M. Naoufal and N. M'barek, **Content Based Image Retrieval Based on Color String Coding and Genetic Algorithm** in *2020 1st International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET)*, 2020, pp. 1–5.
18. R. C. Gonzalez, R. E. Woods, and S. L. Eddins, ***Digital image processing using MATLAB*.** Pearson Education India, 2004.
19. A. C. Bovik, ***The essential guide to image processing*. Academic Press**, 2009.
20. A. R. Smith, **Color gamut transform pairs**, *ACM Siggraph Comput. Graph.*, vol. 12, no. 3, pp. 12–19, 1978.
21. C.-H. Su, H.-S. Chiu, and T.-M. Hsieh, **An efficient image retrieval based on HSV color space**. in *2011 International Conference on Electrical and Control Engineering*, 2011, pp. 5746–5749.
22. L. G. Shapiro and G. C. Stockman, ***Computer vision*.** Prentice Hall, 2001.
23. R. M. Haralick, K. Shanmugam, and I. Dinstein, Textural **Features for Image Classification**. *IEEE Trans. Syst. Man. Cybern.*, vol. SMC-3, no. 6, pp. 610–621, 1973, doi: 10.1109/TSMC.1973.4309314.
24. S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, **Optimization by simulated annealing**. *Science (80-.).*, vol. 220, no. 4598, pp. 671–680, 1983.
25. A. Dekkers and E. Aarts, **Global optimization and simulated annealing**. *Math. Program.*, vol. 50, no. 1–3, pp. 367–393, 1991.
26. J. H. Holland, ***Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence***. MIT press, 1992.
27. K. V. Madhavi, R. Tamilkodi, and K. J. Sudha, **An innovative method for retrieving relevant images by getting the top-ranked images first using interactive genetic algorithm**. *Procedia Comput. Sci.*, vol. 79, pp. 254–261, 2016.

28. M. E. ElAlami, **A novel image retrieval model based on the most relevant features**. *Knowledge-Based Syst.*, vol. 24, no. 1, pp. 23–32, 2011.
29. R. Ashraf, K. Bashir, A. Irtaza, and M. T. Mahmood, **Content based image retrieval using embedded neural networks with bandletized regions**. *Entropy*, vol. 17, no. 6, pp. 3552–3580, 2015.