RBIR USING INTEREST REGIONS AND BINARY SIGNATURES

Thanh The Van (Hue, Vietnam)

Thanh Manh Le (Hue, Vietnam)

Dedicated to András Benczúr on the occasion of his 70th birthday

Communicated by János Demetrovics

Abstract. In this paper, we introduce an approach to overcome the low accuracy of the Content-Based Image Retrieval (CBIR) (when using the global features). To increase the accuracy, we use Harris-Laplace detector to identify the interest regions of an image. Then, we build the Region-Based Image Retrieval (RBIR). For the efficient image storage and retrieval, we encode images into binary signatures. The binary signature of an image is created from its interest regions. Furthermore, this paper also provides an algorithm for image retrieval on S-tree by comparing the images' signatures on a metric similar to EMD (earth mover's distance). Finally, we evaluate the created models on COREL's images.

1. Introduction

It is difficult to find images in a large set of them. A solution is to label the images [1, 2] but it is costly, time-consuming and unfeasible for many applications. Moreover, the labeling process depends on the semantic description of the image. So, an image retrieval system which is based on content is developed to extract visual attributes for the description of image content [3]. Some

digital image retrieval systems have been built, such as QBIC, ADL, DBLP, Virage, Alta Vista, SIMPLIcity, etc.

In recent years, some papers regarding CBIR are published, such as image retrieval based on color histogram [1, 2], image retrieval on the base of binary bit string and S-tree [3], extracting objects on the image based on the change of histogram value [5], the similarity image retrieval based on the comparison of feature regions and their similarity relationship on image [6], color image retrieval based on the detection of local regions by Harris-Laplace method [7], color image retrieval based on bit plane and $L^*a^*b^*$ color space [8], changeable color space to query the content of color images [9], etc.

This paper approaches the semantic description of image contents through binary signatures and stores them on an S-tree. The S-tree data structure describes the relationship between binary signatures. So, it describes the relationship in the contents of images. Based on the description of the semantic relationship of image contents of the S-tree data structure, the paper finds out the similarity images by the content on COREL images [12].

The paper builds the similarity image retrieval on the basis of the local interest regions. First of all, it extracts the interest regions using Harris-Laplace method. Then, the paper creates interest regions for the image. Based on these regions, it creates binary signatures and evaluates the similarity of images. In order to speed up the query, the paper presents S-tree to store binary signatures so as to build a similarity image retrieval algorithm on the S-tree. The paper devotes two main sections to reduce the amount of storage space and to speed up the query of image objects on a large image database.

The binary signature of images is used to describe the content of an image. Comparing to GCHs (Global Color Histograms) and CCVs(Color-Coherence Vectors), the binary signature method in VBA (Variable-Bin Allocation) saves over 75% and 87.5% in storage space, respectively [3].

2.   Data structure and similarity measure

2.1.   Binary signature

In accordance with [4], the binary signature is formed by hashing the data objects, and it has k 1 bits and $(m - k)$ 0 bits in the bit chain $[1..m]$, where m is the length of the binary signature. The data objects and the object of the query are encoded with the same algorithm. When the bits in the signature data object cover completely the ones in the query signature, the data object is

a candidate for the query. There are three cases: (1) the data object matches the query: each bit in s_q is covered with the ones in the signature s_i of the data object (i.e., $s_q \wedge s_i = s_q$); (2) the object does not match the query (i.e., $s_q \wedge s_i \neq s_q$); (3) the signatures are compared and give a *false drop* result.

2.2. EMD distance

Suppose that I is a set of suppliers, J is a set of consumers, c_{ij} is the transportation cost from supplier $i \in I$ to consumer $j \in J$, we need to find out flows f_{ij} to minimize the total cost $\sum_{i \in I} \sum_{j \in J} c_{ij} f_{ij}$ with the constraints [10, 11]: $f_{ij} \geq 0, \sum_{i \in I} f_{ij} \leq y_j, \sum_{j \in J} f_{ij} \leq x_i, i \in I, j \in J$, where x_i is the provider's general ability $i \in I$, y_j is the total need of the consumer $j \in J$. The feasibility condition is

$$\sum_{j \in J} y_j \leq \sum_{i \in I} x_i.$$

The EMD distance [10, 11] equals

$$EMD(x, y) = \frac{\sum_{i \in I} \sum_{j \in J} c_{ij} f_{ij}}{\sum_{i \in I} \sum_{j \in J} f_{ij}} = \frac{\sum_{i \in I} \sum_{j \in J} c_{ij} f_{ij}}{\sum_{j \in J} y_j}.$$

Each image in the database is quantized into a fixed number of n colors $c_1, c_2, ..., c_n$. Each color c_j is represented by a bit string of length m, i.e. $b_1^j b_2^j ... b_m^j$ for $1 \leq j \leq n$, so each image can be described as a sequence of bits

$$S = b_1^1 b_2^1 ... b_t^1 ... b_1^n b_2^n ... b_m^n.$$

The binary signature of image I is $SIG_I = B_I^1 B_I^2 ... B_I^n$, where $B_I^j = b_1^j b_2^j ... b_m^j$, $b_i^j \in \{0, 1\}$. The weight of component B_I^j equals $w_I^j = w(B_I^j) = \sum_{i=1}^{m} (b_i^j \times \frac{i}{m} \times 100)$. Therefore, we have a weight vector of image I as $W_I = \{w_I^1, w_I^2, ..., w_I^n\}$. Let J be the image whose similarity we should calculate corresponding to the image I. We need to minimize the cost to convert the color distribution $\sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} f_{ij}$, where $F = (f_{ij})$ is a color distribution flow matrix between c_I^i and c_J^j. Let $D = (d_{ij})$ be a Euclidean distance matrix in the RGB color space between c_I^i and c_J^j.

We set

$$W_m = \min\left(\sum_{i=1}^{n} w_I^i, \sum_{j=1}^{n} w_J^j\right) \quad \text{and} \quad W_M = \max\left(\sum_{i=1}^{n} w_I^i, \sum_{j=1}^{n} w_J^j\right).$$

The distance $EMD(I, J)$ is the color distribution flows from an image with the color weight as W_M to an image with color weight as W_m. We have the similarity measure between two images I and J with the following formula:

$$EMD(I, J) = \min_{F=(f_{ij})} \frac{\left(\sum\limits_{i=1}^{n} \sum\limits_{j=1}^{n} d_{ij} f_{ij} \right)}{\sum\limits_{i=1}^{n} \sum\limits_{j=1}^{n} f_{ij}},$$

with $\sum\limits_{i=1}^{n} \sum\limits_{j=1}^{n} f_{ij} = W_m = \min \left(\sum\limits_{i=1}^{n} w_I^i, \sum\limits_{j=1}^{n} w_J^j \right).$

2.3. S-tree

S-tree [3, 4] is a tree with many balanced branches. Each node of the S-tree contains a number of pairs $\langle sig, next \rangle$, where sig is a binary signature and $next$ is a pointer to a child node. The root of the S-tree contains at least two pairs and at most M pairs $\langle sig, next \rangle$. All internal nodes in the S-tree can accommodate at least m and at most M pairs $\langle sig, next \rangle$, $1 \le m \le M/2$. The leaves of the S-tree contain the image's binary signatures sig, along with unique identification oid for those images. The maximal height of the S-tree for n signatures is $h = \log_m n - 1$.

Building the S-tree is done by *inserting* and *splitting*. The S-tree only contains a null leaf at the beginning. Then, we insert signatures into the S-tree. When the node v is full, we split into two nodes. At the same time, the parent node v_{parent} is created (if it does not exist). Simultaneously, two new signatures are inserted into node v_{parent}.

The algorithm creates the S-tree that stores the image's binary signature based on the EMD distance as follows:
Algorithm1. Create the S-tree from the set of signatures
Input: $S = \{\langle sig_i, oid_i \rangle | i = 1, ..., n\}$
Output: The $Stree$.
1: *Step 1.*
2: $v = Root$;
3: **if** $S = \emptyset$ **then** STOP;
4: **else**
5:     Choosing $\langle sig, oid \rangle \in S$; $S = S \backslash \langle sig, oid \rangle$;
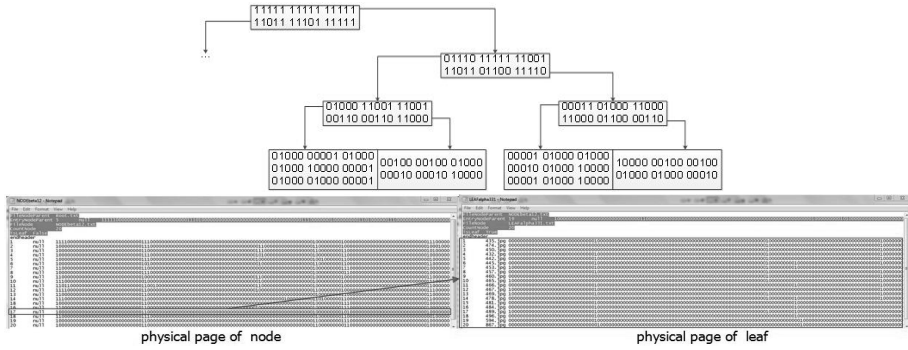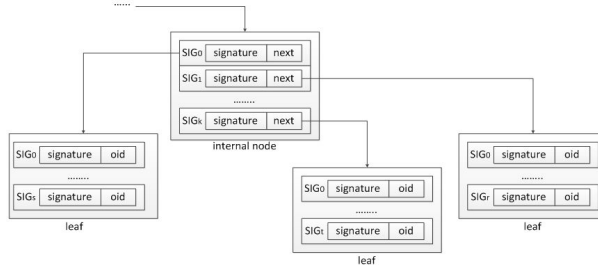6:   Go to Step 2.
7: **end if**

Figure 1. A sample of S-tree

Figure 2. A sample of nodes and leaves in S-tree

8: *Step 2.*
9: **if** v is $leaf$ **then**
10:    $v = v \cup \langle sig, oid \rangle$;
11:    UnionSignature(v);
12:    **if** $v.count > M$ **then**
13:       SplitNode(v);
14:       To go back Step 1;
15:    **end if**
16: **else**
17:    $EMD(SIG_0 \rightarrow sig, sig) = \min\{EMD(SIG_i \rightarrow sig, sig) | SIG_i \in v\}$;
18:    $v = SIG_0 \rightarrow next$;
19:    To go back Step 2;
20: **end if**
21: **return** $Stree$

Procedure1. Union of binary signatures at node v
Input: Node v
Output: The $S - tree$ after union of signatures
 1: **procedure** UNIONSIGNATURE(v_{parent})

2:    $s = \bigcup sig_i$, with $sig_i \in v$;
3:    **if** $v_{parent}! = null$ **then**
4:      $SIG_v = \{SIG_i | SIG_i \to next = v, SIG_i \in v_{parent}\}$;
5:      $v_{parent} \to (SIG_v \to sig) = s$;
6:      UnionSignature(v_{parent});
7:    **end if**
8: **end procedure**

*Algorithm*1 creates the S-tree from the signature file S. Each signature $sig \in S$ is inserted into the most appropriate leaf. If the leaf is full, it would be split. Then, the S-tree grows in the direction of the root.

The S-tree has many balanced branches. Each node of the S-tree is traversed in the best similarity measure direction with EMD distance, the height of the S-tree is $h = \lceil \log_m n - 1 \rceil$. So, the cost of the query process on S-tree is $k \times M \times \lceil \log_m n - 1 \rceil$, where k is the length of each signature, m is the minimum number of signatures, M is the maximum number of signatures of a node in the S-tree. However, if the appropriate leaf is full, it would be split based on $\alpha - seed$, $\beta - seed$ operations ([4]) and the similarity measure EMD, which is done as follows:

Procedure2. Split node v
Input: a node v
Output: the $Stree$ (after splitting the node v)

1: **procedure** SPLITNODE(v)
2:    Create the nodes v_α and v_β contain $\alpha - seed$ and $\beta - seed$;
3:    **for** $SIG_i \in v$ **do**
4:      **if** $EMD(SIG_i \to sig, \beta - seed) < EMD(SIG_i \to sig, \alpha - seed)$ **then**
5:        $v_\alpha = v_\alpha \cup SIG_i$;
6:      **else**
7:        $v_\beta = v_\beta \cup SIG_i$;
8:      **end if**
9:    **end for**
10:    $s_\alpha = \bigcup sig_i^\alpha$, with $sig_i^\alpha \in v_\alpha$;
11:    $s_\beta = \bigcup sig_i^\beta$, with $sig_i^\beta \in v_\beta$;
12:    $v_{parent} = v_{parent} \cup s_\alpha$;
13:    $v_{parent} = v_{parent} \cup s_\beta$;
14:    **if** $v_{parent}.count > M$ **then**
15:      SplitNode(v_{parent});
16:    **end if**
17: **end procedure**

3. Image retrieval

3.1. Extracting interest regions

In order to extract the visual features of images, we standardize the image size (i.e. converting input image in different sizes into an image with the size of $k \times k$). After that, we extract the color feature of images. Because the image based on JPEG standard is described on the YCbCr color space, we need to use YCbCr to extract the features of the images.

According to [7, 8], the Gaussian transformation by human's visual system is given by

$$L(x,y) = \frac{1}{10}[6.G(x,y,\delta_D) * Y + 2.G(x,y,\delta_D) * Cb + 2.G(x,y,\delta_D) * Cr]$$

with

$$G(x,y,\delta_D) = \frac{1}{\sqrt{2\pi}.\delta_D}.\exp\left(\frac{x^2+y^2}{2.\delta_D^2}\right).$$

The intensity $I_0(x,y)$ for color image is calculated with the equation

$$I_0(x,y) = Det(M(x,y)) - \alpha.Tr^2(M(x,y)),$$

where $Det(\bullet), Tr(\bullet)$ are the Determinant and the Trace of the matrix, $M(x,y)$ is a second moment matrix which can be defined as

$$M(x,y) = \delta_D^2.G(x,y,\delta_I) * \begin{bmatrix} L_x{}^2 & L_xL_y \\ L_xL_y & L_y{}^2 \end{bmatrix},$$

where δ_I, δ_D are the integration scale and differentiation scale, and L_α is the derivative computing the α direction. The interest points of the color image are extracted from the formula $I_0(x,y) > I_0(x',y')$ with $x', y' \in A$ and $I_0(x,y) \geq \theta$, where A is the neighborhood of point (x,y) and θ is a threshold value. Let $O_I = \{o_I^1, o_I^2, ..., o_I^n\}$ be a set of interest circles, with the centers as the interest points and the set of interest radiuses as $R_I = \{r_I^1, r_I^2, ..., r_I^n\}$. The values of interest radiuses are in $[0, \min(M,N)/2]$, where M, N are the height and the width of the image. They are extracted by LoG method (Laplace-of-Gaussian). For each image, the process of extracting interest points is described as follows:

Algorithm2. Extract region-of-interest in image
Input: Image I, threshold θ, scales δ_I, δ_D
Output: Interest region of image $O_I = \{o_I^1, o_I^2, ..., o_I^N\}$

Figure 3. A sample of interest regions

1: *Step 1.* For each pixel $p \in I$, convert from RGB color space $(R, G, B \in [0, 1])$ to YCbCr color space as follows:

2:
$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 65.481 & 128.553 & 24.996 \\ -37.797 & -74.203 & 112 \\ 112 & -93.786 & -18.214 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix}$$

3: *Step 2.* Perform Gaussian transformation compiled with the human visual system to calculate the $L(x, y)$ as follows:

4: $L(x, y) = \frac{1}{10}[6.G(x, y, \delta_D) * Y + 2.G(x, y, \delta_D) * Cb + 2.G(x, y, \delta_D) * Cr]$

5: *Step 3.* Calculate the feature intensity $I_0(x, y)$ for color images as follows:

6: $I_0(x, y) = Det(M(x, y)) - \alpha.Tr^2(M(x, y))$

7: *Step 4.* Collect the set of interest points P as follows:

8: $P = \{p(x, y) \in I | I_0(x, y) > I_0(x', y') \wedge I_0(x, y) \geq \theta, (x', y') \in A\}$

9: *Step 5.* Implement the extraction interest regions $O_I = \{o_I^1, o_I^2, ..., o_I^n\}$ based on the set of interest points P.

10: **return** $O_I = \{o_I^1, o_I^2, ..., o_I^n\}$

3.2. Creating binary signature of an image

For each interest region $o_I^i \in O_I$ of an image I, the histogram is calculated on the basis of the standard color range C. Effective clustering method relied on the Euclidean measure in RGB color space classifies the colors of each pixel on the image. Let p be a pixel of image I, so we have a color vector in RGB as $V_p = (R_p, G_p, B_p)$. Let $V_m = (R_m, G_m, B_m)$ be a color vector of the set of standard color range C, so $V_m = \min\{||V_p - V_i||, V_i \in C\}$. At that time, pixel p can be standardized in accordance with color vector V_m. According to experiment, the paper uses the standard color range on MPEG7 to calculate histogram for color images on COREL database. We let $o_I^i \in O_I$ $(i = 1, ..., N)$ be the interest circles of the image I, the histogram vector of the circle o_I^i is $H(o_I^i) = \{H_1(o_I^i), H_2(o_I^i), ..., H_n(o_I^i)\}$. If we let $h_k(o_I^i) = \frac{H_k(o_I^i)}{\sum_j H_j(o_I^i)}$, a standardized histogram vector will be $h(o_I^i) = \{h_1(o_I^i), h_2(o_I^i), ..., h_n(o_I^i)\}$. The binary signature describes $h_k(o_I^i)$ as $B_I^k = b_I^1 b_I^2 ... b_I^m$ with $b_I^j = 1$ if $j = [(h_j(o_I^i) + 0.05) \times m]$, otherwise $b_I^j = 0$. So, the signature describes the interest region $o_I^i \in O_I$ as $Sig(o_i^I) = B_I^1 B_I^2 ... B_I^n$. For this reason, the binary signature of the image I is

$S_I = \bigcup\limits_{i=1}^{N} Sig(o_I^i)$. The process of creating binary signatures for color images is described as follows:

Algorithm3. Creating binary signature of image I
Input: Image I, the look-up table of color vectors $C = (V_1, V_2, ..., V_n)$
Output: Binary signature S_I of image I

1: *Step 1.*
2: $O_I = \text{Algorithm2}(I, \theta, \delta_I, \delta_D)$; (i.e. $O_I = \{o_I^1, o_I^2, ..., o_I^N\}$)
3: Initialize binary signature $S_I = B_1^0 B_2^0 ... B_n^0$, with $B_j^0 = b_1^0 b_2^0 ... b_m^0$, $b_i^0 = 0$,
    $i = 1, ..., m$, $j = 1, ..., n$.
4: *Step 2.*
5: Initialize histogram vector $H(o_I^i) = \{H_1(o_I^i), ..., H_n(o_I^i)\}$
6: **for** pixel $p \in o_I^i$ **do**
7:     Calculate color vector $V_p = (R_p, G_p, B_p)$
8:     $V_m = \min\{||V_p - V_i||, V_i \in C\}$
9:     $H_m(o_I^i) = H_m(o_I^i) + 1$
10: **end for**
11: *Step 3.*
12: Initialize vector $h(o_I^i) = \{h_1(o_I^i), ..., h_n(o_I^i)\}$
13: **for** $h_k(o_I^i) \in h(o_I^i)$ **do**
14:     $h_k(o_I^i) = \frac{H_k(o_I^i)}{\sum\limits_{j} H_j(o_I^i)}$

15: **end for**
16: *Step 4.*
17: **for** $h_k(o_I^i) \in h(o_I^i)$ **do**
18:     **for** i=1..m **do**
19:         **if** $j = \left[(h_j(o_I^i) + 0.05) \times m\right]$ **then**
20:             $b_I^j = 1$
21:         **else**
22:             $b_I^j = 0$
23:         **end if**
24:     **end for**
25:     $B_I^k = b_I^1 b_I^2 ... b_I^m$
26: **end for**
27: $Sig(o_i^I) = B_I^1 B_I^2 ... B_I^n$.
28: *Step 5.*
29: $S_I = S_I \vee Sig(o_I^i)$
30: $O_I = O_I \backslash \{o_I^i\}$
31: **if** $O_I \neq \emptyset$ **then**
32:     To go back *Step 2.*
33: **else**

34:      **return** S_I
35: **end if**

Algorithm3 creates a binary signature of image I. At Step1 based on Algorithm2, it extracts the interest regions $O_I = \{o_I^1, o_I^2, ..., o_I^N\}$. At Step2, it calculates the histogram vector $H(o_I^i) = \{H_1(o_I^i), ..., H_n(o_I^i)\}$ based on interest regions $o_I^i \in O_I$ with the set of standard color C. At Step3, it standardizes vector $h(o_I^i) = \{h_1(o_I^i), ..., h_n(o_I^i)\}$. At Step4, it creates the binary signature for $h_k(o_I^i)$ as $B_I^k = b_I^1 b_I^2 ... b_I^m$ with $b_I^j = 1$ if $j = \left[(h_j(o_I^i) + 0.05) \times m\right]$, otherwise $b_I^j = 0$. At that time, the signature describes the interest region $o_I^i \in O_I$ as $Sig(o_i^I) = B_I^1 B_I^2 ... B_I^n$. At Step 5, it creates the binary signature of image I as $S_I = \bigcup\limits_{i=1}^{N} Sig(o_I^i)$.

3.3.   Image retrieval algorithm

After storing the signatures and identifications of the corresponding images on the S-tree, the query process finds out the binary signatures of similarity images on the basis of traversing S-tree. After finding the image's signatures and relying on the identifications of images, we can find out a set of similarity images corresponding to a query image. For this reason, the problem needs to find out the signatures of the images and the corresponding identifications. This query process is performed by the following algorithm:

Algorithm4. Query image in S-tree
Input: query signature sig and $S - tree$
Output: Set of image signatures and identifications
 1: $v = root$; $SIGOUT = \emptyset$; $Stack = \emptyset$;
 2: Push(Stack, v);
 3: **while not** Empty($Stack$) **do**
 4:      $v = $ Pop($Stack$);
 5:      **if** v **is not** $Leaf$ **then**
 6:          **for** $SIG_i \in v$ and $SIG_i \to sig \wedge sig = sig$ **do**
 7:              $EMD(SIG_0 \to sig, sig) = \min EMD(SIG_i \to sig, sig)|SIG_i \in v\}$;
 8:              Push(Stack, $SIG_0 \to next$);
 9:          **end for**
10:      **else**
11:          $SIGOUT = SIGOUT \cup \{\langle SIG_i \to sig, oid_i \rangle | SIG_i \in v\}$;
12:      **end if**
13: **end while**
14: **return** $SIGOUT$;

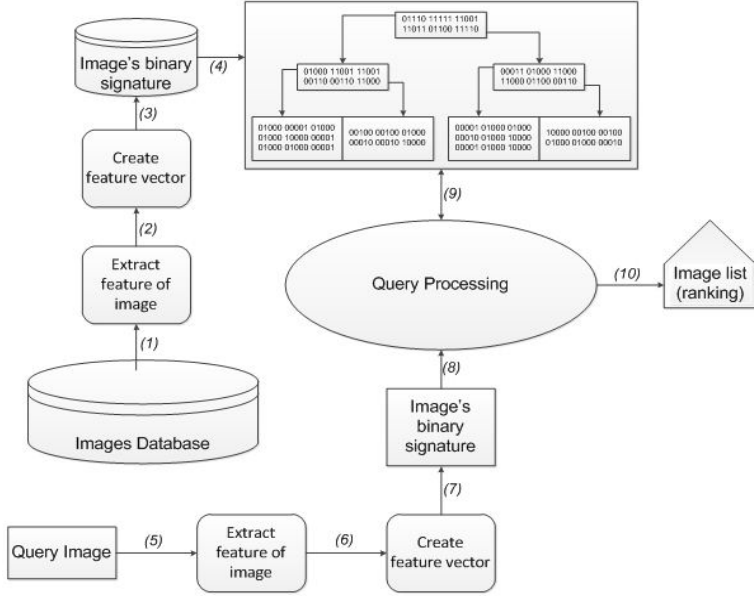The searching process is done similarly to traversing the S-tree. So, the

Figure 4. Model of the region-based image retrieval system

cost of the query process on the S-tree is $k \times M \times \lceil \log_m n - 1 \rceil$ where k is the length of each signature, m is the minimum number of signatures, M is the maximum number of signatures of a node in the S-tree.

4.   Experiments

4.1.   Image retrieval model

The experimental process includes two phases. The first one performs pre-processing to convert image data into binary signatures and puts them into the S-tree based on the similarity measure EMD. The second one performs query process corresponding to a query image which can be converted into a binary signature and queried on the S-tree on the basis of similarity measure EMD. After finding the signatures of similarity images, we find out similarity images and arrange them with similarity measure EMD.

Phase 1: *Perform pre-processing*

Step 1. Extract the features of images in the database to form feature
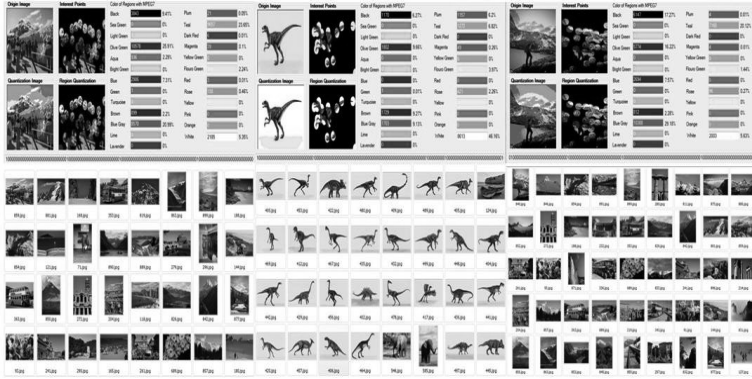
Figure 5. Some results of image retrieval system

vectors.

Step 2. Convert feature vectors of images into binary signatures.

Step 3. Calculate in turn the EMD distance of image's signatures and insert them into the S-tree.

Phase 2: *Perform query*

Step 1. With each query image, we extract the feature vector and convert it into binary signature.

Step 2. To implement the binary signature retrieval process on S-tree including image's signatures, we find out similarity images at each leaf node of the tree through the similarity measure EMD.

Step 3. After having similarity images, we arrange them in the order of the similarity from high to low level and give the list of images. Then, we arrange them on the basis of similarity measure EMD.

4.2. Experimental results

The experimental process is queried on COREL sample data [12] including 10,800 images which can be divided into 80 different subjects. With each query image, we retrieve images on COREL data so as to find out the most similar ones to the query image. Then, we can compare to a list of the subjects of images to evaluate the accuracy.
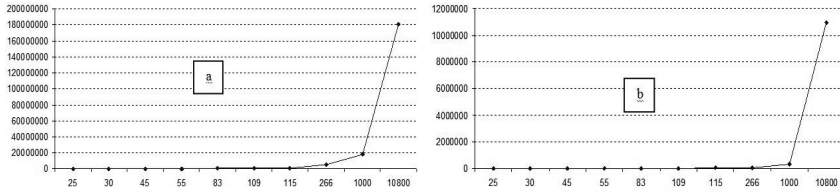
Figure 6. (a)-number of operators to create S-tree; (b)-the time (milliseconds) to create S-tree
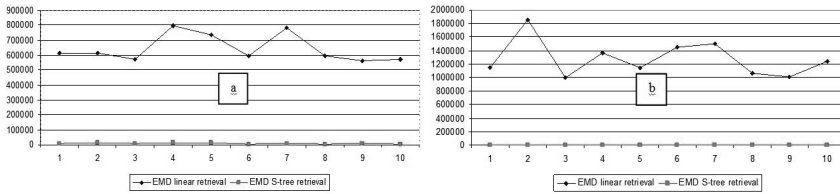

Figure 7. (a)-number of operators to query image; (b)-the time (milliseconds) to query image
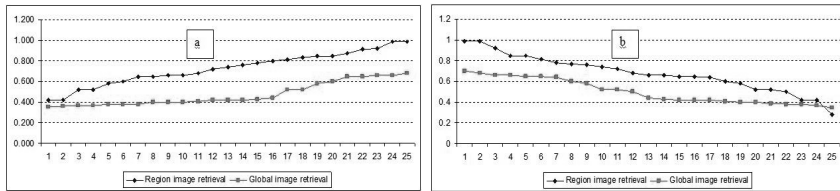

Figure 8. (a)-Recall; (b)-Precision

5.   Conclusion

The paper gives the evaluation method of similarity between two images on the basis of binary signatures. At the same time, it simulates the experimental application on classification of image data of COREL. The experiment shows the evaluation method based on binary signatures and S-tree which can speed up querying similarity images as quickly as possible compared with the linear query. However, the use of feature color gives inaccurate results with the meaning of image's contents. Therefore, the next development extracts the feature objects on images. So, it builds a binary signature to describe objects as well as the content of images. At the same time, it creates a data structure to describe the relationship on the basis of similarity measure between the images.

References

[1] **Neetu Sharma, S., S. Paresh Rawat and S. Jaikaran Singh,** Efficient CBIR using color histogram, *Signal & Image Processing*, **2** (2011), 94–112.

[2] **Fazal Malik, Baharum Bin Baharudin,** Feature analysis of quantized histogram color features for content-based image retrieval based on Laplacian filter, *Int. Conf. on System Engineering and Modeling*, **34** (2012), 44–49.

[3] **Nascimento, M.A., E. Tousidou, V. Chitkara and Y. Manolopoulos,** Image indexing and retrieval using signature trees, *Data & Knowledge Eng.*, **43** (2002), 57–77.

[4] **Chen, Y. et al.,** On the signature tree construction and analysis, *IEEE Trans. Knowl. Data Eng.*, **18** (2006), 1207–1224.

[5] **Liu, G.H. and J.Y. Yang,** Content-based image retrieval using color difference histogram, *Pattern Recognition*, **46** (2013), 188–198.

[6] **Bartolini, I., P. Ciaccia and M. Patella,** Query processing issues in region-based image databases, *Knowl. Inf. Syst.*, **25** (2010), 389–420.

[7] **Wang, X.Y. et al.,** Robust image retrieval based on color histogram of local feature regions, *Multimed. Tools Appl.*, **49** (2010), 323–345.

[8] **Wang, X.Y. et al.,** Robust color image retrieval using visual interest point feature of significant bit-planes, *Digital Signal Processing*, **23** (2013), 1136–1153.

[9] **Tang et al.,** Robust image hash function using local color features, *Int. J. Electron. Commun. (AE)*, **67** (2013), 717–722.

[10] **Rubner, Y., C. Tomasi and L.J. Guibas,** A metric for distributions with applications to image databases, it IEEE Int. Conf. on Computer Vision, Bombay, India, 1998, 59–66.

[11] **Bahri Abdelkhalak and Hamid Zouaki,** EMD similarity measure and metric access method using EMD lower bound, *Int. J. of Computer Science & Emerging Technology*, **2** (2011), 323–332.

[12] **Corel Corp,** `http://www.corel.com`

Thanh The Van
Faculty of Information Technology
Hue University of Sciences, Hue University
77 Nguyen Hue str.
Hue, Vietnam
`vanthethanh@gmail.com`

Thanh Manh Le
Hue University
03 Le Loi str.
Hue, Vietnam
`lmthanh@hueuni.edu.vn`