When image indexing meets knowledge discovery

Chabane Djeraba
IRIN, Ecole Polythechnique de l'Université de Nantes,
2 rue de la Houssinière, BP 92208 - 44322 Nantes Cedex 3, France
E-mail : djeraba@irin.univ-nantes.fr

ABSTRACT
In our paper, we deal with the challenge of extending automatically the classic image indexing by visual relationship features. The visual relationship features are discovered automatically from images. They contribute to make more efficient the content-based indexing. More particularly, we develop an advanced content-based indexing articulated around the following notions : - classic indexing, - clustering algorithm, - visual feature book and relationship qualification.

Keywords
Image, Indexing, retrieval, content, similarity, knowledge discovery, relations.

1. INTRODUCTION
In large image databases, finding images that contain semantic content, such as flowers during autumn or goals during football plays, is not simple. To do so, images should be well annotated by experts when inserted in the database. So, the quality of retrievals depends on the quality of the manual annotations. This solution characterizes classic information retrieval systems initiated by [Moo 51], and developed by [Sal 68], [Rij 79], and others. However, manual annotations tend to be incomplete and inconsistent, and they do not allow visual content-based image indexing and retrieval. Visual information systems, also known by content-based indexing and retrieval systems, such as in [Dje 00], [Jai 98] and others, overcome some of these shortcomings. The index is, generally, created automatically, and the final users have the possibility to formulate content-based queries. In spite of these appreciable advantages, the automatic indexing, which is the most important advantage of visual information systems, support weak semantic description, and therefore weak semantic queries. So finding images that contain flowers during autumn remains a very difficult query.

Content-based image indexing associated to knowledge discovery may be seen as a new way of thinking and regarding retrieval of multimedia information and it opens up to a lot of new applications which have not been possible, previously. For image archives the new possibilities given by content-based image indexing and knowledge discovery lies in the ability to perform "advanced queries-by-example", meaning that we can present an image of an object, pattern, texture, etc., and fetch the images in the database that most resemble the example of the query. For image databases the new possibilities lie in the ability to access efficiently and directly selected images of the database.

Our paper deals with the following challenge : how do we build automatically the semantic content of images, based on basic content descriptions ? We believe that discovering hidden relations among basic features contributes to extract semantic descriptions useful to make the content-based image retrieval more efficient. In our case, the relationship discovery are held into two important steps : symbolic clustering based on the new concept of visual feature book and relevant relationships discovery.
The originality of our work concerns the following points :

- the definition of a new algorithm of global/local clustering and classification, based on : - visual quantization, powerful image descriptors and - suitable similarity measures,
- the creation of an efficient feature (texture, color) book which is the most representative of database image features,
- the power qualification of the relationship among visual features. They are composed of conditional probability and implication intensity measures,
- the extension of the classic indexing by relevant relationships that are automatically discovered.

The implementation of these notions together in the same framework constitute our advanced content-based indexing which is the scope of the paper.
We organize the paper as follow : in section 2, we describe the classic and advanced content based indexing and retrieval. We answer to the following questions : how images are searched in image database. We will not focus on speed data structures necessary to support the index, however, we will focus on the knowledge necessary to advanced content-based retrieval. In section 3, we present how the content of images are extracted and represented, how descriptors of images may be used to discover

Proceedings of the International Workshop on Multimedia Data Mining (MDM/KDD'2000), in conjunction with ACM SIGKDD conference. Boston, USA, August 20, 2000
(O.R. Zaïane, S.J. Simoff, eds.)

relations between descriptors, and how the discovered relations are useful to content-based image retrieval. In section 4, we describe some experiment results.

2. YOU SAID CONTENT-BASED IMAGE INDEXING AND RETRIEVAL ?

The content-based image indexing and retrieval architecture is composed of three important components : extraction, representation and retrieval. Extraction and representation components constitute the heart of the architecture, together, they constitute the indexing component. The extraction component extract, automatically or semi-automatically, regions in images and compute features such as color, texture and shape of these regions. The whole image may constitute itself a region. The extracted contents are represented as or transformed into suitable models and data structures, and then stored in a persistent index.

The retrieval component constitute the eyes of the architecture. It searches images by selecting target images or content properties such as color, sketched shape, texture of image regions, or combinations of these. The retrieval process computes distances between source (example) and target features, and sorts the most similar images.

The central question is : how to extract and represent the content in order to make the retrieval process efficient ? Before answering this question, we will start by presenting the classic approach, and we will compare the benefits of the knowledge discovery to image indexing and retrieval efficiency.

2.1 Classic indexing

Indexing responds to how the content should be extracted and represented to allow efficient and effective search and access ?

Sequential searching of images with simple similarity computations is quite appropriate in a small database. However, the larger the database is, the slower the sequential approach is. So efficiency will not be respected. Classic access structures such as B-trees [Bay 72], K-D trees [And 85], point quadtrees [Fin 74] and R-trees [Gut 84] have advantages and disadvantages. Point quadtrees are simple to implement. However, there is a complexity of both insertion and search. Furthermore, deletion in point quadtrees is complex because finding a candidate replacement node for the node being deleted is generally difficult. Finally, the range retrieval in point quadtrees is time consuming. It takes $O(2\sqrt{n})$, where n is the number of image references in the tree. K-D-trees are very simple to implement. However, the search and insertion complexity in k-d-tree is high. In MX-quadtrees, range retrieval is very efficient, and the insertion, deletion and search take time proportional to O(n). We assume that the image (ex. map) is split up into a grid of size (2^n x 2^n) cells. R-trees have been preferred over k-d trees and point quadtrees, because they store a large number of rectangles in each node. So, they are suitable for disk accesses by reducing the height of the tree, this leading to fewer disk accesses. The disadvantage of R-trees is that, in certain cases, instead of following one path in the search process, multiple paths may be followed, because bounding rectangles associated with different nodes may be overlapped. Multiple paths means more disk accesses that might be compared to disk accesses of the other quadtrees.

These representations are physical access structures, they deal with applications that require massive amounts of storage and disk accesses. So they concern low level representation of the access structures. These access structures are necessary, but not enough to access effectively image materials. They need to be completed by high level representations (logical representation) that organize efficiently the descriptors of images, independently of their physical representations.

2.2 Advanced indexing

To obtain efficient access data structures, we should combine physical and logical representations of high-dimensional features. In our context, to effective up the content-based retrieval, we consider semantic representations that include image class hierarchy (images of flowers, panorama, etc.) characterized by knowledge and access speed data structures (K-D-trees). The K-D trees are implemented for high-dimensional features, at least eleven-dimension color and texture attributes, and voluminous classes. However sequential search is used for low-dimensional features and less voluminous classes. The K-D-trees are implemented at eleven-dimension because the color is represented by one dimension and the texture is represented by ten dimensions (ten couple of coefficients).

For example, when the user asks for images that contain waterfalls (figure 1), the system matches the user examples with the knowledge in the form of rules. In certain case, the image may belong to several classes, because the distance between the gravity center of the examples and the knowledge of the image classes are near together. In all cases, the retrieval process focuses its matches in the sub-classes of the current ones. In the sub-class, it triggers the same match process. When the leaf class is reached, the physical data structure is used to find the best images. When the number of the images in a class is low (ex. less than 100), than the search process is limited to sequential order.

In the example presented bellow, the first images returned contain waterfall, and the other images contain flowers. The whole images are visually similar to the example images. This example illustrates the « advanced query by examples » that is based on combination of visual features (texture and color) and knowledge. «advanced query by examples» specifies a query that means «find images that are similar to those specified». The query may be composed of several images. Several images accurate the quality of retrieval. For example, Several images of a «waterfall» accurate the description of the waterfall. This property makes possible the refinement of retrieval based on the feed backs (results of previous queries).

In the retrieval task (figure 2), features (colors, textures) of the query specification are matched with the knowledge associated to classes (ex. natural, people, industries, etc.). The suited classes are « Natural », then the matching process focus the search on the sub-classes of Natural : « Flowers », « Mountain », « Water », « Snow », etc. The knowledge associated to flowers and waterfalls are verified, so the matching process focuses the search on the « Flower » and « Water » classes. « Flowers » and « Water » classes are leaves, so the matching process compares the features of the examples with features of the image database to determine which images are similar to the example features. The matching task is based on computing the distance between target and source

image regions. When mixing several features, such as colors and textures, the resulting distance is equal to the Sum taking into account the ponderation values of the considered features. The resulting images are sorted, the shortest distance corresponds to the most similar images.

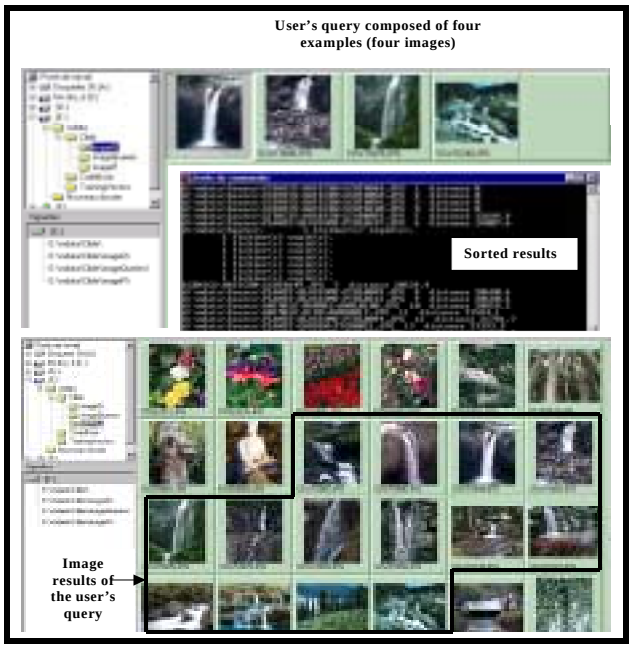

Figure 1 : «find images that contain waterfalls».

An important advantage of the advanced indexing is the efficiency of the content-based retrieval. When the user gives examples of image to formulate his query, and asks "find images similar to the examples", the system will not match the source image with all the images in the database. It will match the source image features with only the target image features of suited classes. If the knowledge associated to a class is globally verified, then the considered class is the suited one. Then, the system will focus the search on the sub-classes of the current one. In the target classes that contain few instances, the search is limited to sequential accesses. Another advantage is the richness of descriptions contained in the results of queries since the system presents both similar images and their classes.

2.3 New architecture

The advanced approach for content-based image indexing needs an advanced architecture. The advanced architecture extends the classic architecture by knowledge in the form of simple rules. Simple rules that characterize each semantic class (flowers, natural, mountain, etc.) are automatically extracted. The classic indexing is base exclusively on low level representations of images and physical access structures, without any knowledge and logical representations of the content. The rules describe relationships between visual features (colors and textures of images). Each set of rules associated to a class summarizes image contents of the class. Rules contribute in the discrimination of each class, so they represent knowledge shared by the classes. When images are inserted in the database, it is classified "automatically" in the class hierarchy. At the end of the classification process, the image is inserted in a specific class. In this case, the distance between the image and the knowledge associated to the class is the shortest one, compared to the distance between the image and the other classes. Otherwise, the instantiation relationship between the image and the class, will not be considered.

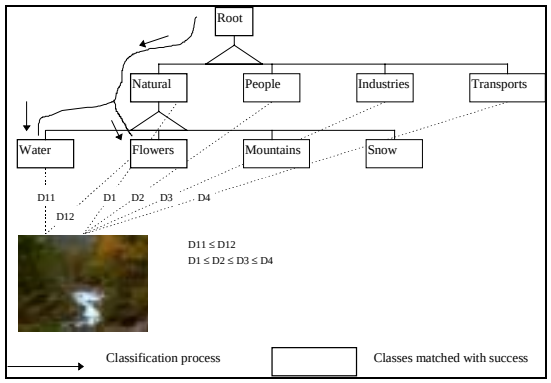

Figure 2 : Example of image insertion into the class hierarchy

This architecture avoids efficient retrievals and browsing through classes. For example, the user may ask "find images similar to the source image but only in People classes" or "find me all images that illustrate the bird class with such colors and such shapes".

3. DISCOVERY HIDDEN RELATIONS

Based on image content description, the knowledge are discovered. The discovered knowledge characterizes visual properties shared by images of the same semantic classes (Birds, Animals, Aerospace, Cliffs, etc.).

The discovery is held into two steps : symbolic clustering and relationship discovering and validation.

1- symbolic clustering

2- relationship discovery and validation

In the first step, numerical descriptions of images are transformed into symbolic form. The similar features are clustered together in the same symbolic features. Clustering simplifies, significantly, the extraction process. For example, in the figure presented bellow (figure 3), the image is composed of region1 and region2. Region1 is characterized by light red color, and region2 by water color and water texture.

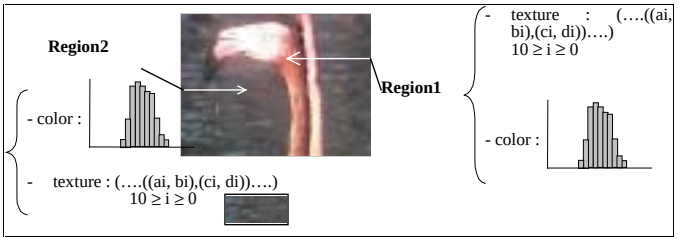

Figure 3 : Original representation of the image. Numeric representation of image B8169

Light red color is not described by a simple string, but by a color histogram. Even if the region colors of different images of the same class, as presented in figure 4, are similar (i.e. light red), the histograms (numerical representation of color) associated with them are not generally identical.

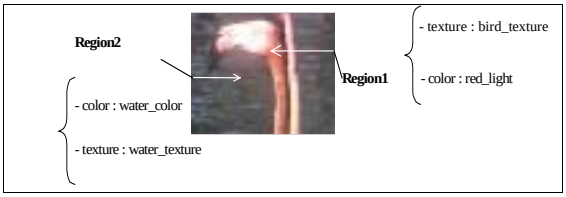

Figure 4 : Symbolic description of image B8169

```
/* Declaration of composition relations between images and regions. */
   is_composed_of(imageB8169,       [region1, region2]).
/* Region features declaration. A region is usually described by texture and color */
/* text attributes. */
   features(region1,                [texture, bird_texture], [color, red_light]]).
   features(region2,                [[texture, water_texture], [color, water_color]]).
/* Image features declaration. An image is usually described by the texture, color. */
features(imageB8169, [[text, text1]]).
```

In the second step, the knowledge discovery engine automatically determines common features between the considered images in rule form. These rules are relationships in the form of `Premise => Conclusion` with a certain accuracy. These rules are called statistical as they accept counter-examples.

```
(texture, water_texture) => (color, water_color) (CP 100%, II 96.08%)
(texture, waterfall_texture) => (color, white_color) (CP 100%, II 87.43%)
(texture, texture_bird) => (color, red_light) (CP 100%, II 40.45%)
```

Before presenting the algorithm of discovering, we will present how the image content (color, texture) are represented and extracted automatically. More details about image descriptors have been presented in [Dje 00].

3.1 Image descriptors

3.1.1 Color

The color is the first descriptor of image content. The color feature is extracted automatically from an image or a region. In the first step of the extraction process, based on a physical format, the region or image color is extracted and represented in the RGB model. Based on the RGB model, the color is transformed into HSV model, characterized by three means H, S and V. The HSV model is more suited than the RGB model, in which certain ambiguities appear between colors (ex. Yellow and Green).

In the object-oriented modeling, we define a class of colors called HSV. HSV class includes color histogram and methods (ex. distance measures). The color of a region is represented by a histogram of 256 colors. Each element of the histogram represents the number of pixels that have the suited color (see figures 5, 6). So, comparing the colors of two regions is equivalent to compute the distance between the histogram of the target and the source regions. Before submitting the query, the user may choice the distance, by default quadratic distance is activated.

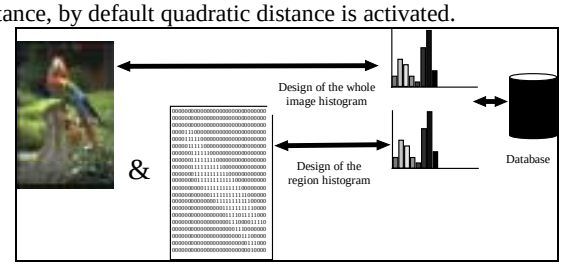

Figure 5 : Extraction of colors.

In figure 5, the color is represented by a histogram. One histogram represents the color of the whole image, and other histograms represent image region colors. An image region is designed by a binary mask. For example, the binary mask designs the image region that characterizes the bird. The binary mask is equal to 1 inside the region, and 0 outside the region. The histogram of colors are calculated on the basis of the binary mask and the photo.

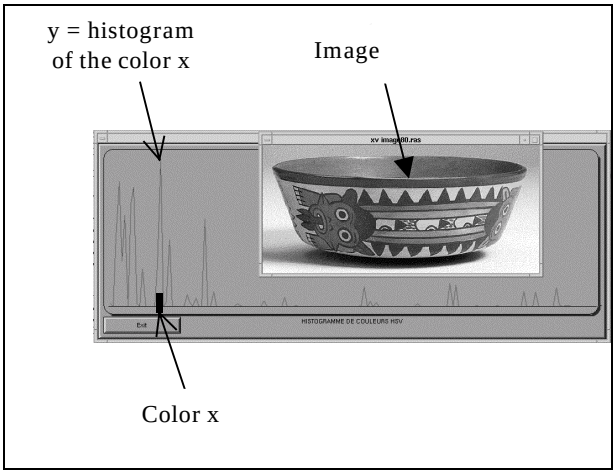

Figure 6 : Color histogram.

In figure 6, the graphic representation of the image color histogram is displayed. For example, y is the histogram of the color x ⇔ y = number of pixels that have the color x.

3.1.2 Texture

The texture is an important aspect of human visual perception, and it is the second important feature extracted automatically from image regions.

When two patterns differ only in scale, the magnified one is coarser. The variance measures the dispersion of the difference of gray-level with a certain distance. The contrast measures the vividness of the texture and is a function of the gray-level

difference histogram. The directionality measures the « peakedness » of the distribution of gradient directions in the image. For example the region may have a favored direction. It is not a powerful texture representation, but may be interesting for retrieval process when mixing it with color features.

The approach, considered, implements a powerful texture representation. Thus, we use a mathematical model which is one of the best : Fourier model [Zah 72]. Fourier model has very interesting advantages : - the texture can be reconstructed from the descriptors. – it has a mathematical description rather than a heuristic one. - And finally, the model supports the robustness of description to translation, rotation and scale transformations. An important contribution of our representation is our extension of Fourier model to texture description. This extension considers the matching process. In this extension, we consider texture(t) composed of two functions : x(t) and y(t).
So texture(t) =(x(t), y(t)). x(t) represents the different level of gray of x, and y(t) represents the different level of gray of y. t indicates the different indices of the signal texture. t = 0, N-1. N is the period of the function, and N = number of x values and y values = length of the normalized image. So, we have two suites of coefficients $S(a_n, b_n)$ and $S(c_n, d_n)$ that represents Fourier coefficients of x(t) and y(t) respectively.

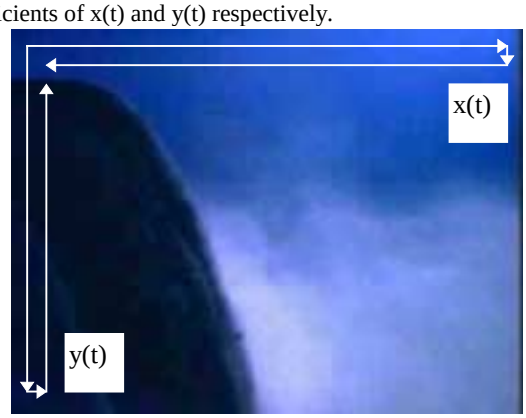

Figure 7 : x(t), y(t)

$$x(t) = a_0 + \sum_{k=1,N} a_n \cos(2\pi kt/N) + b_n \sin(2\pi kt/N)$$

$$y(t) = b_0 + \sum_{k=1,N} c_n \cos(2\pi kt/N) + d_n \sin(2\pi kt/N)$$

and

$$a_n = 2/N \sum_{k=1,N} x(t) \cos(2\pi kt/N)$$

$$b_n = 2/N \sum_{k=1,N} x(t) \sin(2\pi kt/N)$$

$$c_n = 2/N \sum_{k=1,N} x(t) \cos(2\pi kt/N)$$

$$d_n = 2/N \sum_{k=1,N} x(t) \sin(2\pi kt/N)$$

Figure 8 : Fourier Coefficients formulas

We consider only eleven coefficients of Fourier that select the lowest frequencies of the sub-band $k \in [0\text{-}10]$. In this extension, we modify the similarity measures (Euclidean distance) in order to consider the coefficients of the two signals x(t) and y(t), as we will see in the following section.

3.2 Symbolic clustering algorithm

The clustering of numeric features in symbolic form raises several problems. The first problem is that a feature may belong to one or several symbol(s). The problem is the same for texture and color features. The second problem is a consequence of the first one. After the symbol creation, we can obtain two different symbols that may be either composed of the same numerical features (equal symbols), or composed of several symbols that differ on only one feature. If we obtain two different symbols composed of the same features, the system keeps only one symbol among symbols composed of the same features. If we obtain several symbols that differ on only one numerical feature, then, it is more difficult to resolve. The problem is the same for the other features. The third problem is that the system generates a symbolic feature base bigger than the numeric feature base since the system computes for one fact containing numeric values, several facts containing symbolic values. The figure presents a part of a symbolic feature and illustrates the possibility of feature fact explosion.
To resolve these problems, we implemented a technique that clusters numerical representation of color, texture, by using data quantization of colors and textures, we use also the term of feature book creation. The color and texture clustering algorithms are similar, the difference is situated in the distance used.

3.2.1 Principle of the algorithm

The algorithm is a classification approach based on the following observation. The scalar quantification of Lloyd developped in 1957 is valide for our vectors (color histogram, fourier coefficients), four rate distribution and for a large variety of distortion criteria. It generalizes the algorithm by modifying the feature book iteratively. This generalization is known by k-means [Lin 80]. The objective of the algorithm is to create a feature book, based on automatic classifications themselves based on a learning set. The learning set is composed of feature vectors of unknown probability density. Two steps should be distinguished :

- A first step of classification that clusters each vector of the learning set around the initial feature book that is the most similar. The objective is to create the most representative partition of the vector space.

- A second step of optimization that permits the correct adaptation in a class of the feature book vector. The gravity center of the class created in the previous step is computed.

The algorithm is reiterated in the new feature book in order to obtain a new partition. The algorithm converges to stable position by evolving at each iteration the distortion criteria. Each application of the iteration of the algorithm should reduce the mean distortion. The choice of the initial feature book will influence the local minimum that the algorithm will achieve, the global minimum corresponds to the initial feature book. The creation of the initial feature book is inspired of the splitting technique [Gra 84].
The splitting method decomposes a feature book Y_k into two different feature books $Y_{k-\varepsilon}$ and $Y_{k+\varepsilon}$, where ε is a random vector of weak energy, and its distortion depends of the distortion of the

splited vector. The algorithm is then applied to the new feature book in order to optimize the reproduction vectors.

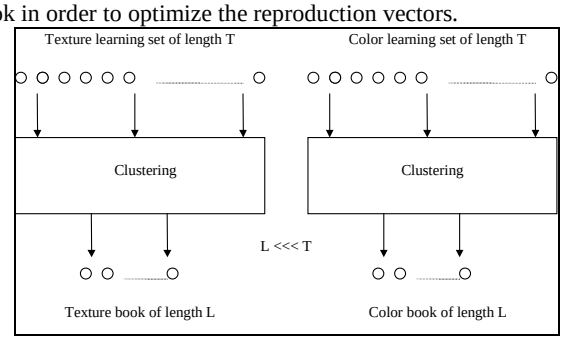

Figure 9 : Clustering and reduction algorithm. In our experiments T = 30.000 and L = 256

3.2.2 Distances

The system clusters similar colors together in a symbolic form by using a suitable distance. In our case, for the color, we implement the quadratic distance which is one of the most accurate distances.

$$D_Q(H,I)=\sqrt{(H-I).A(H-I)^T} \text{ or } D_Q^2(H,I)=\sum_{p=1}^{n}\sum_{q=1}^{n} a_{pq}\left(h_{c_p}-i_{c_p}\right)\left(h_{c_q}-i_{c_q}\right)$$

With A: the similarity matrix (n×n), $A=[a_{pq}]$, a_{pq}: weight of the similarity between the p and q bins

Figure 10 : Quadratic_distance definition.

This distance takes into account the color similarity between the histogram bins by using the symmetrical similarity matrix *A*. The matrix weights may be normalized to obtain $0 \le a_{pq} \le 1$. So, the matrix diagonal is equal to 1, since any color is identical with itself (a_{pp}=1). A coefficient a_{pq} close to 0, represents a dissimilarity between *p* and *q* bins. For example, in QBIC, the quadratic distance between two color histograms, is used with a similarity matrix *A* whose elements are defined by [Haf 95]: $a_{ij} = (1 - d_{ij}/d_{max})$, with $d_{max} = max_{ij}(d_{ij})$, d_{ij} being Euclidean distance between the color i and j in any color space. The two distributions *H* and *I*, may also be normalized in order that $0 \le h_{c_p}, i_{c_p} \le 1$ *and* $\sum_p h_{c_p} = 1 = \sum_p i_{c_p}$.

$$D_{L2}(H,I)=\sqrt{\sum_{i=1}^{n}(h_{c_i}-i_{c_i})^2}$$

Figure 11 : L2-distance or Euclidean distance definition.

This distance makes it possible to obtain satisfactory results since it appreciates color similarity correctly. However, its major drawback is that it is time-consuming compared to the other distances. Euclidean distance results from the quadratic distance where *A* matrix is the identity matrix (no correlation between the histogram bins).

In our example, the light red color zones in the different images are grouped together in the symbolic form red_light as they are similar. Water color in not clustered in red_light, because the distance between them is not short enough. However, it is clustered in the symbolic form water_color shared with other images. In the same way and based on appropriate distances, the system clusters respectively similar shapes, similar textures together in a symbolic form.

For the texture, we implement an adaptation of the Euclidean distance to Fourier coefficients, we call it « texture_Fourier_distance ». So, the matching distance between the Fourier descriptors of the texture *t'* of an image image' and the Fourier descriptors of the texture *t* of an image « image », is triggered by computing the distance between *t* and *t'*, namely: $d(t,t') = \sqrt{(\sum_{n=1,N} (|T'_n - K.|T_n|)^2)}$, N=10, for t and t' textures, we have a positive constant K, and for any $n \neq 0$, $|T'_n| = K*|T_n|$, where $Z_n = \sqrt{(|X_n|^2 + |Y_n|^2)} = \sqrt{(a_n^2 + b_n^2 + c_n^2 + d_n^2)}$

That is to say, the textures are identical near to one geometric transformation. The translation, scale and rotation have no effect on the module of Fourier coefficients. $K = 1/N*(\sum_{n=1,N}(|T'_n|/|T_n|))$ is an estimation of *K* which minimizes the error on the *N* (e.g. 11) first coefficients of Fourier.

3.2.3 Algorithm

Based on the learning set of length equal to T, the algorithm finds a feature book of colors and textures of length equal to L, that are the most representative colors and textures of image databases.

Global Clustering

FeatureBook Y_f = SymbolicClustering (visual feature = VisualFeature, learning set = LearningSet, Y_0, T, L)

{

if the VisualFeature = color then LearningSet = {H_1, H_2, H_3, ..., H_T}, a set of T histograms.

If VisualFeature = texture then LearningSet = {S1, S2, ..., S_T}, a set of T sequence of Fourier coefficients. Y_0 is the initial feature book with distortion D_0 and cardinal equal to L.

Pre-conditions : L << T

Invariant : s ≤ S=L/2

1- Initialization : D_0 = Distortion (Y_0) ; E_0 = Entropy(Y_0) ; s = 0 ; s = number of splitting activated. $Class_0$ = {$Class_{0,k}$; k = 1, ..., L}

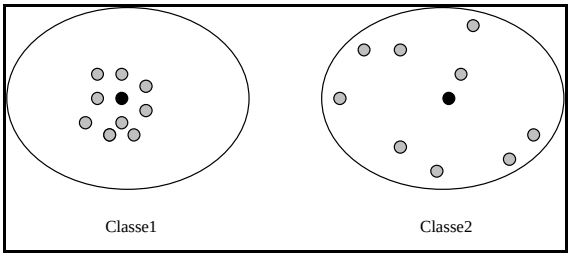

Figure 12 : Distorsion(Classe1) < Distorsion(Classe2)

While (s < S)

{

2 - s = s + 1

3 - Splitting of the VisualFeature of the feature book Y_{s-1} that support the highest apparition probability p_i. p_i corresponds to the $class_{s,i}$ that has the maximum number of instances. The VisualFeature of the feature book corresponds to the gravity center of the $class_{s,i}$. $(Y_{s-1,i'}, Y_{s-1,i''}) = splitting(Y_{s-1,i})$.

4 - Deletion of the VisualFeature of the feature book Y_{s-1} that support the lowest apparition probability p_j. p_j corresponds to the $class_{s,j}$ that has the minimum number of of instances. The VisualFeature of the feature book corresponds to the gravity center of the $class_{s,j}$.

Each splitting is followed by a deletion, so the cardinal of the feature book remains constant (equal to L).

5 - A local clustering with the parameter E_1 is executed on the class $class_{s,i}$ on the local feature book composed of $Y_{s-1,i'}$, $Y_{s-1,i''}$ and E_1 the stop criteria of the algorithm.

Y_s = Clustering(visual feature = VisualFeature, feature book = ($Y_{s-1,i'}$, $Y_{s-1,i''}$), E_1, learning set = $class_{s,i}$).

6 - A global clustering is executed on the global feature book composed of Y_s with the parameter E_2. E_2 is the stop criteria of the algorithm.

Y_s = Clustering(visual feature = VisualFeature, feature book = Y_s, E_2, learning set = $class_s$) ;

D_s = Distortion (Y_s) ;

E_s = Entropy(Y_s).

$D_s < D_0$: the distortion is reduced and $H_s > H_0$: the entropy is augmented}}

Ideally, the stop criteria of the algorithm should depend of the distortion D_s, however, the distortion D_s depends of the number of splitting.

Local clustering

FeatureBook Y_f = Clustering(visual feature = VF, learning set = LS, Y_0, Y_f, T, L, E)

{

Y_0 is the initial feature book with distortion D_0 and length equal to L. LS is the learning set with a length is equal to L. E is the stop criteria.

Pre-conditions : L << T

1 - Initialization : D_0 = Distortion (Y_0) ; s = 0 ; s = number of splitting activated.

Do

{

2 - Based on the feature book $Y_s = \{Y_{s,k}\ k=1,..,L\}$ and the learning set LS; we extract the partition $Class_s = \{Class_{s,k}\ ; k = 1, ..., L\}$, in which distance(x, y) is minimal. So :

$x_t \in Class_{i,k}$ when distance(x_t, y_k) ≤ distance(x_t, y_j) ∀ j ≠ k.

$D_s = 1/T \sum_{t=1,T} \min_Y$ distance(x_t, y), y ∈ Y_s

if VF = texture then distance = texture_fourier_distance, presented bellow.

if VF = color then distance = quadratic_distance, presented bellow.

3 - Creating the optimal catalogue $Y_{s+1} = \{centroid(Class_{s,k})\ k=1,..,L\}$; $centroid(Class_{s,k})$ corresponds the gravity center of the class $Class_{s,k}$. $centroid(Class_{s,k}) = (1/|Class_{s,k}|)* \sum x_t / t : x_t \in Class_{s,k}$. $|Class_{s,k}|$ is the number of instances in $Class_{s,k}$.

4 - s = s + 1

} Until (D_{s-1} - D_s)/D_s < E}

The distortion D_s is a positive and decreasing function. Each iteration of the algorithm reduce the distortion. So, $D_{s-1} \geq D_s$.

The experimental results showed that the distortion values decrease quickly compared to splitting evolution. After the quick decreasing, the distortion values decrease very slowly. Conversely, The entropy increase quickly compared to splitting evolution, and then, it increases very slowly.

3.3 Relationship discovery and validation

Based on the feature book, the discovery engine is triggered to discover the shared knowledge in the form of rules, and this constitutes the second step the general algorithm.

Accuracy is very important in order to estimate the quality of the rules induced. The user should indicate the threshold above which rules discovered will be kept (relevant rules). In fact, the weak rules are rules that are not representative of the shared knowledge. In order to estimate the accuracy of rules, we implement two statistical measures : conditional probability and implication intensity. The conditional probability formula of the rule a => b makes it possible to answer the following question: ''what are the chances of proposition b being true when proposition a is true ? The definition of this measure is P(b/a) = Card(A∩ B)/Card(A)

More intuitively, conditional probability allows us to estimate the accuracy of a rule, considering the number of counter-examples. For example, let us consider p_1 (a => b) and p_2 (b => a) conditional probabilities are respectively 100% and 5.6%. So, the rule b =>a has a lot of counter-examples. In *E* (universe set), there are lots of objects that belong to B, but not to A. Conversely, the rule a => b has no counter-example. So, objects that respect proposition a, respect also proposition b.

Conditional probability allows the system to determine the discriminating characteristics of considered images. Furthermore, we completed it by the intensity of implication [Gra 82]. For example, implication intensity requires a certain number of examples or counter-examples. When the doubt area is reached, the intensity value increases or decreases rapidly contrary to the conditional probability that is linear. In fact, implication intensity simulates human behavior better than other statistical measures and particularly conditional probability. Moreover, implication intensity increases with the considered population sample representativity. The considered sample must be large enough in

order to draw relevant conclusions. Finally, implication intensity takes into consideration the sizes of sets and consequently their influence. For example, conditional probability of a => b is P_1 (100%) and implication intensity of a =>b is φ_1 (23%) values are very different because conditional probability does not take into consideration the fact that proposition b is verified by lots of objects. On the contrary, implication intensity considers that it is not surprising that an object of *A* verifies proposition b because proposition b is verified by many objects of the considered sample.

Let A, B and E sets respectively be the sets of instances that verify proposition a, the set of instances that verify proposition b, and the set of all instances or the universe set. From a theoretical point of view, implication intensity measures the degree of statistical astonishment of size $A \cap \overline{B}$ (this set contains objects that verify proposition a and that do not verify proposition b) considering the sizes of *A*, *B* and E sets, and assuming there is no a priori link between A and B. The cardinals or the sizes of A and B subsets of E are determined by the objects of the database belonging to A and B.

The knowledge discovery engine returns the rules in the form of Premise => Conclusion whose intensity and conditional probability are greater than or equal to a certain threshold. For the moment, this threshold is defined manually (ex. 90 %). Samples of extracted rules by the prototype are (texture, water_texture) => (color, water_color), (texture, waterfall) => (color, white) with respective conditional probability values of 100% and 100%, and implication intensity values of 96.08% and 87.08 %.

3.4 Some comments

The set of induced rules corresponds to knowledge shared by classes. This knowledge is helpful for user's comprehension of the class. Extracted rules are validated when the conditional probability and the rule intensity are greater than a special value (i.e. 90% for conditional probability and 80% for implication intensity). For example, (texture, bird_texture) => (color, red_light) has 100% conditional probability and 40.4598% implication intensity. Since the rule intensity is less than 80%, the system will not store it. We explain this weak measure of rule intensity by the fact that there are few examples that respect this rule.

In our example, the searched class is characterized by a set of rules such as rule 1. So, if we have the ''water_texture'' texture in an image of the class, then the region color inside the image is red_light with 100% conditional probability and 96,08 % rule intensity. So, during image database creation, the classification of an image in a class is possible if the class rules, previously extracted and validated, are globally respected. At least 50 % of rules are respected. If not, we will not consider the instantiation relationship between the image and the class.

x => y has 15.3846% conditional probability and 61.79% implication intensity, that is to say that the conditional probability value is less than 90%. So, the system did not store this rule. We explain this weak measure of conditional probability by the fact that there are a lot of counter-examples of the considered rule.

(texture, waterfall) => (color, white) is a good rule because the conditional probability value is 100% and the implication intensity is 81.79%. This rule means that when we have a texture that includes water, then we would have a white region color.

In the retrieval task, when the user specifies an image (called source image) as the basis of his query, and asks ''find images similar to the source image'', the system will not match the source image with all the images of the database. It will match the source image features with all the target images of the appropriate classes. These classes contain rules globally respected by the source image.

For example, if we have a source image that contains a ''texture_waterfall'', but it does not globally verify the rules associated with this concept, we can deduce the weakness of the relationship between the source image and the class. The system matches the source image with classes through their rules stored in the database.

4. EXPERIMENTAL RESULTS AND CONCLUSION

We have conducted extensive experiments of varied data sets to measure the performance of the advanced content-based query.

The recall and precision graphic for our system are computed as follows. References («query») of images are selected from a test collection. A sub-set of images is selected per class (waterfalls, fires, panorama, etc.). For each image, a knowledge content-based query is formulated. For an image reference, we associate a knowledge content-based query that includes visual features (color, texture, color + texture). We also associate a classic content-based query that uses classic indexing (there is no knowledge integration).

To demonstrate the efficiency of the knowledge content-based queries, the results of the advanced content-based queries are compared with the results of queries that do not use classic content-based queries. Since it is not possible to retrieve all relevant images, our experiment evaluates only the first ranked images.

Judging on the results, it is obvious that the use of knowledge leads to improvements in both precision and recall over majority queries tested. The average improvements of advanced content-based queries over classic content-based queries are 23% for precision and 17 % for recall. Precision and recall are better for concept-based queries (queries that mix visual features and textual descriptions with different degrees of importance) than for queries that use only visual features such as color or shapes or textures or textual descriptions, but not both.

Acknowledgement

Many thanks to Henri Briand for his encouragement.

REFERENCES

[And 85] Andrew W. Appel, An Efficient Program for Many-Body Simulation, SIAM Journal of Statistical and Scientific Computing, 6(1), January 1985.

[Bay 72] Bayer, R., E. McCreight. Organization and Maintenance of Large Ordered Indexes. Acta, 1972, Informatica 1(3), 173-189.

[Dje 00] Djeraba C., Bouet M., Henri B., Khenchaf A. « Visual and Textual content based indexing and retrieval », to

appear in International Journal on Digital Libraries, Springer-Verlag 2000.

[Fay 96] Fayyad U. M., Piatetsky-Shapiro G., Smyth P., Uthurusamy R., «Advances in Knowledge Discovery and Data Mining», AAAI Press, MIT Press, 1996.

[Gut 84] Antonin Guttman: R-Trees: A Dynamic Index Structure for Spatial Searching. SIGMOD Conference 1984, pages 47-57, 1984.

[Gra 82] Gras Régis, THE EISCAT CORRELATOR, EISCAT technical note, Kiiruna 1982, EISCAT Report 82/34, 1982.

[Gra 84] Gray R. M. « Vector Quantization », IEEE ASSP Mag., pages 4-29, April 1984.

[Gup 97] Amarnath Gupta, Ramesh Jain «Visual Information Retrieval», A communication of the ACM, May 1997/Vol. 40, N°5.

[Haf 95] Hafner J., al. «Efficient Color Histogram Indexing for Quadratic Distance Functions». In IEEE Transaction on Pattern analysis and Machine Intelligence, July 1995.

[Jai 98] Ramesh Jain: Content-based Multimedia Information Management. ICDE 1998: 252-253

[Lin 80] Linde Y., Buzo A., Gray R. M. « An algorithm for Vector Quantizer Design », IEEE Trans. On Comm., Vol. COM-28, N° 1, pages 84-95, January, 1980.

[Moo 51] Moores C. N. «Datacoding applied to mechanical organization of knowledge» AM. Doc. 2 (1951), 20-32.

[Rag 89] Raghavan, V., Jung, G., and Bollman, P., “A Critical Investigation of Recall and Precision as Measures”, ACM Transactions on Information Systems 7(3), page 205-229, 1989.

[Rap 74] Raphael A. Finkel, Jon Louis Bentley: Quad Trees: A Data Structure for Retrieval on Composite Keys. Acta Informatica 4: 1-9, 1974.

[Rij 79] C. J. Keith van Rijsbergen «Information retrieval», Second edition, London: Butterworths, 1979

[Sal 68] Salton Gerard «Automatic Information Organization and Retrieval», McGraw Hill Book Co, New York, 1968, Chapter 4.

[Zah 72] C. T. Zahn, R. Z. Roskies, « Fourier descriptors for plane closed curves », IEEE Trans. On Computers, 1972.