An Adaptive and Efficient Clustering-Based Approach for Content-Based Image Retrieval in Image Databases

Renato O. Stehling
Institute of Computing
Univ. of Campinas, Brazil
renato.stehling@ic.unicamp.br

Mario A. Nascimento
Dept. of Computing Science
Univ. of Alberta, Canada
mn@cs.ualberta.ca

Alexandre X. Falcão
Institute of Computing
Univ. of Campinas, Brazil
afalcao@ic.unicamp.br

Abstract

In this paper, we present a new Content-Based Image Retrieval (CBIR) approach, for image databases, based on cluster analysis. CBIR relies on the representation (metadata) of images' visual content. In order to produce such metadata, we propose an efficient and adaptive clustering algorithm to segment the images into regions of high-similarity. This approach contrasts with those that use a single color histogram for the whole image (global methods), or local color histograms for a fixed number of image cells (partition-based methods). Our experimental results show that our clustering approach offers high retrieval effectiveness with low space overhead. For example, using a database of 20,000 images we obtained higher retrieval effectiveness than partition-based methods with about the same space overhead of global methods, which are typically regarded to as storage-wise compact.

1 Introduction

Image databases are becoming more and more common in several domains. The evolution of techniques for acquisition, transmission and storage of images have allowed the construction of very large image databases. This has spurred great interest in the area of content based image retrieval (CBIR). Image content can be described at various levels. It may regard perceptual features (also known as content-dependent metadata) like color, texture, shape, structure and spatial relationship, or semantic primitives (also known as content-descriptive metadata) such as the identification of real-world objects and the meaning of the images [2].

In this paper, we are interested in CBIR techniques suitable to large and heterogeneous image database. We consider heterogeneous a set of images that belong to potentially different domains; have different sizes and resolutions; and have distinct visual content. Indeed, this is a typical scenario on the World Wide Web. When working with large and heterogeneous collections of images, it is not possible to assume any *a priori* knowledge about the visual content of the images. Moreover, it is not possible to use domain dependent metadata, and the cost of using semi-automatic image analysis techniques are prohibitive. Because of these restrictions, our interest is geared toward content-dependent metadata – data which refers to perceptual low/intermediate-level features of the images which can be automatically extracted.

Retrieval of images according to their perceptual low-level features is inherently different and more complex than retrieval of well-structured (traditional) data [14]. For each image, it is necessary to precompute the visual metadata that describes the visual content of the images and that is used to support CBIR. As users usually have an imprecise knowledge about the characteristics of the images, Query-by-Example (QBE) seems to be the most adequate way to submit queries to a CBIR system. A query image specified in QBE is processed to extract the same kind of visual metadata that describes the images in the image database, allowing a direct comparison between the visual metadata that describe two images.

As one can observe, CBIR is not an exact process such as the evaluation of a boolean combination of predicates on a set of attributes. CBIR systems must take into account the uncertainty factor introduced during the query formulation and during the image analysis process, which results in an imprecise description of the image content. Hence, CBIR is strongly tied to the concept of similarity-based retrieval, which is the task of reordering the database images according to their similarity to the query image. It is concerned with ranking rather than classification [2], which is the case in the traditional database scenario. The (dis)similarity between two images is based on a distance function that directly compares the visual metadata of the images.

Ranking algorithms are at the core of CBIR systems and attempt to establish a simple ordering of the retrieved im-

1098-8068/01 $10.00 © 2001 IEEE

ages. Images appearing at the top of this ordering are considered to be more likely to be relevant. Images are presented and examined sequentially by the user in order to decide about their relevance. The relevance of a retrieved image has a degree of psychological subjectiveness. Different users, or even the same user under other circumstances, may perceive the visual content of an image in a different way.

Being this the case, once a new CBIR approach is conceived, it is necessary to evaluate its performance objectively. In traditional DBMS research, the response time and the space requirements are usually the metrics adopted for evaluating a system. In CBIR systems, however, there is the additional issue of evaluating the ranking algorithm according to the relevance of the retrieved images to the user's needs (effectiveness).

Towards that goal, and focusing on the domain of image databases, we present CBC (Color-based Clustering), a new CBIR approach based on cluster analysis. The CBC approach uses an adaptive and agglomerative clustering algorithm to segment images into a set of disjoint connected regions[1], composed by pixels with a predefined degree of color similarity. Our approach is able to represent, and compare, images based on the set of regions in which they are decomposed. The CBC approach has the advantage of avoiding the problem of color-space quantization while also allowing the exploration of alternative types of visual metadata per region, e.g., their spatial location and size.

This paper is organized as follows. Section 2 reviews some related works. Section 3 describes CBC, our clustering-based approach to CBIR. Section 4 reports our experiments to compare the retrieval effectiveness among three variations of our approach and five other existing CBIR approaches and also includes the results of these experiments. Finally, Section 5 presents our conclusions and directions for future work.

2 Related Work

Color is a visual feature which is immediately perceived when looking at an image. The most simple and well known approach to encode the low-level color information of an image is the global color histogram - GCH [7, 20]. A GCH is obtained by quantizing the colors of an image and counting how many pixels belong to each color. Assuming an image composed by n pixels, k being the k^{th} color in an uniformly quantized color-space and $a[k]$ being the number of pixels whose color is k, a GCH h is an array of real numbers, each one describing one quantized color, according to the function $h[k] = a[k]/n$.

[1]Two regions are disjoint if they do not share any pixel. One region A is connected if there is a path between every pair of pixels of A, formed only by pixels of region A [9].

Pass et al [12] proposed another technique based on color histograms called Color-Coherence Vector (CCV). In CCV approach, for each color k in a quantized color-space, there are two values associated: the number of coherent pixels vs. the number of incoherent pixels with this color. A pixel is coherent if it is part of a large similarly-colored connected region. The argument used is that the comparison of coherent and incoherent feature vectors between two images allows for a finer distinction of similarity than when using a simple GCH. Stricker and Orengo [19] proposed the CMM approach. In this approach, instead of representing the complete color distributions, it is represented only their dominant features, via the first three moments of each color channel in the HSV color space.

The approaches described above are all *global methods*, in the sense that they extract the visual metadata of the whole image (globally). Global methods have the advantage to have a compact representation and that the extracted visual metadata can be, under certain constraints, efficiently compared and indexed using a Spatial Access Method [8]. The main disadvantage is that in these approaches, there is no information about the spatial distribution of colors inside the images. Thus, images with very different spatial layout may have similar representations, specially in large collections of images.

In order to take into account the spatial distribution of colors and consequently improve retrieval, several *regional methods* have been proposed. In regional methods, an image is segmented in a set of regions accordingly to a predefined visual property and each region is represented individually. Some known techniques used to segment an image in regions are: (1) spatial partitioning [1, 11]; (2) density estimation [3, 13] and (3) hierarchical clustering [1].

Spatial partitioning methods usually superimpose a grid of cells over the image[11]. We refer to such methods as grid-based approaches. The image content is represented by an array of local color histograms (LCHs), one for each cell, according to the rule: $h[l][k] = a[l][k]/n$, where n is the number of pixels of the image, l is the l^{th} cell of the grid superimposed on the image, k is the k^{th} color in an quantized color-space and $a[l][k]$ is the number of pixels whose color is k in cell l.

A simple variation of the grid-based approach is the Color-Shape Histograms – CSH [17]. In CSH, instead of using a LCH for every cell, a "cell histogram" is used for every color to describe their spatial distribution. The storage space overhead is reduced since, in general, the images contain a relatively low number of colors.

Next, we will discuss two works more related to ours in the sense that they also deal with image segmentation for CBIR: (1) the QBIC project, based on cluster analysis and (2) the Blobworld system, based in parametric density estimation.

QBIC – In QBIC [1], the pixels of an image are clustered in a way that, at the end of the process, the segmented image has only a small number of colors (5 to 15), depending on the image. During the clustering process, two clusters are merged if their mutual rank falls bellow a predefined threshold. The mutual rank of clusters P and Q is $n + m$, where Q is the n^{th} closest cluster to P and P is the m^{th} closest cluster to Q. The distance between two cluster is measured as the Euclidean distance between their mean colors. The clustering process also considers the similarity between the spatial distribution of clusters' pixels. For each final color obtained, the connected components of the pixel population having that color are identified and, for each connected component, a bounding rectangle is calculated. The bounding rectangles for a given color are successively clustered into groups of geometrically close rectangles until one rectangle remains. The result is a hierarchical tree structure for each cluster color. The distance between two regions is calculated as a weighted sum of the distance between the colors themselves and the distance between their associated tree. The distance between two images I and J is the average of the distance between each region of I and its closest region in J. There is no reference to the computational complexity of the approach (clustering procedure and image distance calculation) and to the space overhead of the extracted visual metadata.

Blobworld – The Blobworld system [3] clusters pixels in a joint color-texture-position eight-dimensional space. The joint distribution of color, texture and position features are modeled as a mixture of Gaussians. The Expectation-Maximization (EM) algorithm is used to estimate parameters of this model.

Creating the Blobworld representation of an image involves three steps: (1) selection of an appropriate scale for each pixel and then extraction of color, texture and position for that pixel at the selected scale; (2) grouping of pixels into the resulting 8D feature space modeling the distribution of pixel features as a mixture of Gaussians; and (3) describing the features of each obtained region.

The color of each region is represented as a 500 bins local color histogram (based on L*a*b* color space). To compare the color of two regions, it was used the quadratic distance between their histograms $d^2(x, y) = (x - y)^T A(x - y)$, where $A = [a_{ij}]$ is a matrix of weights representing the similarity between bins i and j. It is unclear from the original paper the space overhead and computational complexity of the approach. It is reported that the segmentation process takes 5-7 minutes per image of size $n = 128 \times 192 = 24,576$ pixels, on a 300MHz Pentium II. The approach we present next (CBC) process an image of size $n = 374 \times 251 = 93,874$ pixels in at most 5 *seconds*, using an 500MHz AMD K6II.

3 Color-based Clustering – Our Approach to CBIR

Our color-based clustering approach to CBIR is based on a fully automatic clustering algorithm which has an efficient implementation. In CBC, each image is decomposed in a set of disjoint connected regions. Each region is larger (in number of pixels) than a threshold size s_0. Additionally, all pixels of a region have a predefined degree of color similarity, according to a threshold color-distance d_0. We denote the procedure to obtain these regions CBC(d_0, s_0), where the thresholds d_0 and s_0 are parameters defined by the user. These parameters have a direct impact on the number of regions in which each image is decomposed. We adopt the convention that parameters d_0 and s_0 are percentages of the maximum value allowed. Thus, $s_0 = 10$ means a threshold size equivalent to 10% of the image size (the maximum possible size of a region) and $d_0 = 5$ means a threshold distance equivalent to 5% of the maximum distance between two points in the chosen color-space. The CBC(d_0, s_0) procedure can be conceptually divided into four main steps:

1. Convert an input image I into a weighted and non-oriented graph $G(V, E)$, where the pixels in I are the vertices of V and each pair (p, q) of 4-adjacent pixels in I define an edge of E whose weight is the Euclidean distance between the colors of p and q in the L*a*b* color-space[2].
2. Compute an adaptive and agglomerative clustering of pixels on G using d_0 as parameter. This algorithm outputs a graph partition where each part is a tree whose nodes form a connected region of pixels and the least similarity between distinct regions is greater or equal to d_0.
3. Merge all regions systematically, whose area is less than s_0, with their most similar neighbor until all regions have area greater or equal to s_0.
4. Characterize each remaining region by a 6D feature vector (L, a, b, s, h, v), where L, a, b are the mean values of the color components L, a, and b in the region, respectively, s is the size (number of pixels) of the region normalized by the image size, and h, v are the normalized horizontal and vertical coordinates of the geometric center of the region in the image.

Note that, the feature vectors of all regions that compose the image I represent its visual content, and so, they are stored in the database. To complete the description of the

[2]The L*a*b* space has been defined in order to make easier the evaluation of perceptual distance between colors. In fact, they are defined according to transformations that approximate a tri-stimulus color-space into an Euclidean space [2].

```
   Single-Linkage($G(V,E), d_0$)
1     for each vertex $v \in V$ do Make-Set($v$)
2     sort the edges of $E$ by nondecreasing weight $w$
3     for each edge $(u,v) \in E$, ordered by nondecreasing weight
4        if $d_{mean}$(Find-Set($u$), Find-Set($v$)) $< d_0$
5           if Find-Set($u$) $\neq$ Find-Set($v$), then Union($u,v$)
6        else break
```

Figure 1. Our implementation of the Single-Linkage algorithm

CBC procedure, we explain in details the clustering algorithm of step 2 next.

3.1 Clustering algorithm

Cluster analysis [5, 10] is one of the most well-developed and commonly used forms of combinatorial data analysis and hierarchical clustering are among the best-known clustering methods. There are two kinds of hierarchical algorithms: *agglomerative* and *divisive*. Agglomerative methods start when all pixels are apart, i.e., they start with n singleton clusters. Then in each step two clusters are merged, until only one is left. On the other hand, divisive methods start when all pixels are together and, in each following step, a cluster is split up, until there are n of them.

Our clustering algorithm consists of a simple and effective variation of the *single-linkage* clustering algorithm. The single-linkage [5, 10] is an agglomerative algorithm which uses the function $d_{min}(A,B) = min(d(A_i, B_j))$ to compute the distance between two clusters $A = \{A_1, A_2, ..., A_n\}$ and $B = \{B_1, B_2, ..., B_m\}$. The clustering process is terminated when the distance between the nearest clusters exceeds a predefined threshold. The single-linkage algorithm is not able to keep two clusters clearly apart when they come very close to each other, because a single link between the two clusters is sufficient to connect them. This characteristic leads to a notorious *chaining effect* [5, 10], by which poorly separated clusters are chained together.

The single-linkage algorithm can be described under the language and concepts of graph theory [5]. Consider the data elements that we want to cluster (image pixels) as being nodes of a graph. The merging of two (initially singleton) clusters $A = \{A_i\}$ and $B = \{B_j\}$ corresponds to adding an edge between the nearest pair of nodes $A_i \in A$ and $B_j \in B$. Since edges are added always between distinct nodes (clusters), the resulting graph never has any closed circuit. In fact, the resulting graph is a *tree*. If this process continues until all clusters are linked together, it can be shown that the resulting graph is a *minimal spanning tree* – MST [5]. Based on the relation between the single-linkage algorithm and minimal spanning trees, we chose to implement our single-linkage algorithm with a implementation equivalent to that of the well known Kruskal's greedy algorithm to generate a MST [4, Section 24.2]. This is the asymptotically fastest implementation known to us. The procedure that describes our variation of the single-linkage algorithm is shown in Figure 1.

The function `Make-Set(.)` creates a cluster with only one element: the node passed as parameter. The function `Find-Set(.)` returns the identifier of the cluster which contains the element passed as parameter. Our clustering algorithm works as follows. Line 1 creates $|V|$ clusters, each with one node of G. The edges in E are sorted by nondecreasing weight in line 2. The `for` loop in lines 3-6 checks if the distance between the clusters of u and v is smaller than the threshold distance d_0 (the stop criteria) for each edge (u,v). If so, and if the clusters are distinct, they are merged. To diminish the effects of the single-linkage chaining effect, we introduced a new heuristic into the stop criteria. Instead of comparing the weight of the edge being analyzed directly with the threshold distance d_0, we compare the distance of the mean vectors of each cluster with d_0. To do this, we assign a mean vector to each cluster and update these vectors whenever a `Union(.,.)` operation is performed. As we can see, we use the d_{min} distance to decide the ordering and which clusters are candidates to be merged, and d_{mean} distance[3] to decide whether or not the candidates should be merged.

This solution has the advantage to be as efficient as the traditional single-linkage procedure and, at the same time, to reduce the chaining effect, the main problem of the single-linkage approach.

[3] $d_{mean}(A,B) = d(A_{mean}, B_{mean})$, where A and B are clusters, and A_{mean} and B_{mean} are A's and B's median (representative) vectors respectively.

```
    d$(A, B, \alpha)$
1   for each pair of regions $A_i \in A$ and $B_j \in B$
2       $A_i.status = 0$
3       $B_j.status = 0$
4       $D_{A_iB_j} = Rd(A_i, B_j, \alpha)$
5   sort the computed distances $D_{A_iB_j}$ in non-decreasing order
6   $\beta = 0$
7   for each distance $D_{A_iB_j}$ in non-decreasing order
8       if $A_i.status = B_j.status = 0$
9           if $A_i.size < B_j.size$
10              $w = A_i.size$
11              $B_j.size = B_j.size - A_i.size$
12              $A_i.status = 1$
13          else
14              $w = B_j.size$
15              $A_i.size = A_i.size - B_j.size$
16              $B_j.status = 1$
17              if $A_i.size = 0$ then $A_i.status = 1$
18          $\beta = \beta + w \times D_{A_iB_j}$
19  return $\beta$
```

Figure 2. Distance function algorithm

Our algorithm differs from Kruskal's algorithm in two ways. First, we do not store the edges that compose the MST. Second, we add a stop criterion (line 4) to finish the process before we obtain only one cluster (the MST). Clearly, the running time of our algorithm is the same as in Kruskal's algorithm: $O(E \log E)$ [4]. Since in our graph model, $E = O(V)$ (because of the connectivity restriction), our clustering procedure runs in time $O(n \log n)$, where $n = |V|$ is the number of pixels in the image.

3.2 Distance Function

In this section, we describe the distance function used to compare two images in the CBC approach. The distance between two images A and B, $d(A, B, \alpha)$, is a weighted composition of the distances between the regions that compose each image – $Rd(A_i, B_j, \alpha)$. Here, A_i and B_j are regions from images A and B respectively, obtained via the clustering algorithm detailed before. The α parameter defines the weights used to combine distances in the color-space with distances between the spatial positions of the regions. The distance function $Rd(A_i, B_j, \alpha)$ between regions A_i and B_j is defined as:

$$\begin{aligned} Rd(A_i, B_j, \alpha) = \quad & \alpha \times L_2(A_i.color, B_j.color) + \\ & (1 - \alpha) \times L_2(A_i.center, B_j.center) \end{aligned}$$

where $L_2(.,.)$ represents the Euclidean distance between its arguments. The computation of the distance between two images $d(A, B, \alpha)$ is algorithmically described in Figure 2.

The distance $d(A, B, \alpha)$ works as follows. Initially, we compute all possible distances between pairs of regions (one of each input image), according to the function $Rd(A_i, B_j, \alpha)$. Additionally, all regions are initialized as "non-matched" (status=0). The initialization steps (lines 1-4) take time $O(nm)$, where n and m are the number of regions in the images A and B, respectively. In line 5, the computed distances are ordered in non-decreasing order. In this way, the first distance value corresponds to the best possible match between a region of image A and a region of image B. The second value corresponds to the second best match and so on. The ordering of nm values takes time $O(nm \log(nm))$. For each distance $D_{A_iB_j}$ in non-decreasing order, we compare the size of the related regions A_i and B_j. The smallest region determines the weight value w that will multiply the value $D_{A_iB_j}$ in order to obtain the distance β between the images. The weight w represents

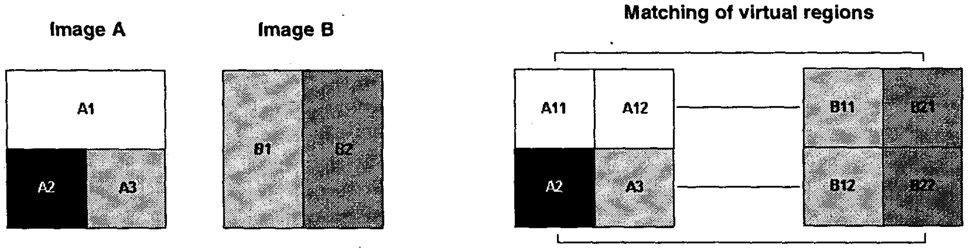

Figure 3. An example of how the function $d(A, B, \alpha)$ decompose real regions into virtual regions

the percentage of the two images that match with distance $D_{A_i B_j}$.

In each iteration of the for loop in line 7, the smallest region related to $D_{A_i B_j}$ is marked as "matched" (status=1). Status equal to 1 means that the smaller region has matched completely, and any other distance related to this region in the next iterations should be discarded. Since at each iteration, the largest region may not be completely matched, we subtract the size of the smallest region (the percentage of the region actually matched) from the size of the largest one. After this operation, if the size of the region equals 0 (occurs when the two compared regions have the same size), then both regions match and we mark them accordingly. If the size of the largest region remains greater than 0, then we continue analyzing distances involving this region until the size of the region equals 0. Observe that the next distances involving a previous analyzed region will always be higher than the previous distance values because we are analyzing these values in non-decreasing order. Observe also that, at the end of the process, the sum of the weight values w equals 1, meaning that the entire image content has been compared. The for loop in line 7 is executed nm times, one for each distance $D_{A_i B_j}$. Thus, the execution of lines [7..18] takes time $O(nm)$.

It is important to note that an actual region A_i of an image may be virtually "broken" into smaller units to find the best possible match for it. The best possible match may involve only one greater region, a set of several smaller regions, or also a combination of partial content of several regions of the other image. In any way, at the end of the process, the "real" regions of the two images were decomposed in possibly smaller "virtual" regions in such a way that: (1) the number of "virtual" regions of the two images is the same, (2) there is a one-to-one correspondence between the virtual regions of two images and (3) each pair of corresponding regions has the same size. Thus, the distance between images becomes a weighted average of the dissimilarities between their corresponding virtual regions where the weights are their sizes.

We will exemplify the application of our distance function using the images in Figure 3. For the sake of simplicity, assume the use of gray-scale images and the distance between two regions given only by the difference of their gray levels. In Figure 3, we have two images A and B that we want to compare. Image A has three regions (obtained via a suitable algorithm, e.g., our proposed CBC): $A = \{A_1 \cup A_2 \cup A_3\}$. Likewise, image B has only two regions: $B = \{B_1 \cup B_2\}$. The best possible match between regions of A and B is (A_3, B_1) because this pair of regions has exactly the same gray level. Since B_1 is larger than A_3, we split B_1 into two virtual regions $B_1 = \{B_{11} \cup B_{12}\}$, in a way that one of the resulting regions (B_{12}) has exactly the same normalized size of region A_3. Thus, we obtain (A_3, B_{12}) and this match will contribute to the distance between A and B with a weight of 0.25 (the normalized size of the matched regions).

The second best match between real regions of images A and B is (A_1, B_1) but, as the region B_1 was split before and one of the resulting regions (B_{12}) was matched in a previous step, then the only possible choice is (A_1, B_{11}). Again, as region A_1 is larger than region B_{11}, the algorithm decomposes A_1 in two virtual regions $A_1 = \{A_{11} \cup A_{12}\}$ in a way that the virtual region A_{12} has the same size of B_{11}. Thus, our second match is (A_{12}, B_{11}) also with weight 0.25. Now, let us assume that the third best match is (A_2, B_2). As B_2 is larger than A_2, the algorithm decomposes B_2 in two regions $B_2 = \{B_{21} \cup B_{22}\}$. The third match is then (A_2, B_{22}) and the final match is (A_{11}, B_{21}). As these regions have the same size, there is no need to split regions and the process is finished. Since all the virtual regions have the same size, the four virtual matches (evaluated according to their gray level distance) are averaged with the same weight $w = 0.25$ in order to obtain the distance between A and B.

4 Experiments

We compared the effectiveness of our CBIR approach with the effectiveness of five other approaches discussed in Section 2: (1) Global Color Histogram – GCH [20], (2) Color-Coherence Vector – CCV [12], (3) Color-Moments

Table 1. Summary of the characteristics of the compared approaches.

Approach	Color-Space	Cells/regions [♯]	Space[♭]	Complexity[‡]	Metric
GCH	RGB	1	64	$O(n)$	L_1
CMM	HSV	1	9	$O(n)$	L_1
CCV	RGB	1	128	$O(n)$	L_1
Grid	RGB	64	4096	$O(n)$	L_1
CSH	RGB	64	1856	$O(n)$	L_1
CBC(5, 0.3)	L*a*b*	11	66	$O(n \log n)$	$d(A,B,\alpha)$[†]
CBC(3, 0.1)	L*a*b*	40	240	$O(n \log n)$	$d(A,B,\alpha)$[†]
CBC(2, 0.05)	L*a*b*	155	930	$O(n \log n)$	$d(A,B,\alpha)$[†]

[♯] Average values for the 20,000 images we used

[♭] Average number of real numbers needed to represent the images in our database

[‡] n is the size (number of pixels) of an image

[†] $\alpha = 0.875$

– CMM [19], (4) Grid (e.g., [11]) and (5) Color-Shape Histograms – CSH [17]. The first three approaches are global methods and the last two are partition-based methods.

We compared the CBC approach with three distinct combinations of threshold color-distance (d_0) and threshold size (s_0). Each combination resulted in a different number of regions per image. Since our distance function depends of a parameter α, we experimentally determined that $\alpha = 0.875$ corresponds to the most adequate compromise. Confirming our intuition, this value suggests that the distance between the colors of the regions is more relevant than the position of such regions when we compare two images. When we used $\alpha = 1$ (eliminating completely the contribution of the spatial location to the distance value), the effectiveness of our approach did become slightly smaller. A summary of the most important characteristics of the eight compared approaches is shown in Table 1.

We evaluated the effectiveness of the approaches in processing image-based queries using a controlled environment. Since we did not count with a population of users, the evaluation study was based on objective criteria. For example, the set of relevant images relative to a given query was determined *a priori*, using an objective relevance criteria in which the relevance of a image relates to how well the image responds to the query that was posed. The subjective view of relevance considers not only the content of an image but also the state of knowledge of the user at the time of the search.

We used a dataset of 20,000 JPEG images from a stock CD by Corel Corp. This database is composed of images of several different domains and, for this reason is called a *heterogeneous database*. In each domain, the images are semantically related, allowing to distinguish one domain from the others easily. Since the images belonging to the same domain may not have a very similar visual content, we called these domains *heterogeneous domains*. We chose 29 of such domains to be used in our experiments. Out of each domain, we selected an image to be used as query image and a set of images visually similar to the chosen query image. These sets were called *relevant result sets* - RRSets.

In any CBIR approach, we expect to retrieve the RRSets as soon as possible, since they correspond to what we consider relevant for each query image. In the average, we have 30 relevant images per query image. We realized two distinct experiments with databases of different sizes. In one experiment, we used the 20,000 images of the whole dataset as a large and heterogeneous database. In the other experiment, we used the union of the RRSets of the first experiment as a small and less heterogeneous database with 1,023 images. We used the same queries and RRSets in both experiments. Our goal was to analyze if the relative performance between the analyzed approaches remains the same when we change the size of database, enlarging/diminishing the proportion of relevant/non-relevant images.

We compared the approaches' effectiveness using two distinct measures: (1) Precision vs. Recall [15] and (2) *Normalized Average Rank* – NavgR' [17] (a derivative of the measure presented in [6]). The effectiveness of a retrieval system is a measure related to the users satisfaction with the system's output. Since it is generally difficult for users to distinguish among a large number of degrees of relevance, we choose to work with binary values: either an image is relevant or it is not. Precision vs. Recall [15] was chosen because it is the best known and widely used measure in information retrieval systems. Although it is not the most adequate measure in the context of ranked output [6, 15, 16], it has been widely used to compare CBIR systems.

The Normalized Average Rank (NavgR) [6] was first used in the QBIC project and then, in some other CBIR approaches [16, 17]. NavgR measures how close the set of relevant items appears to the top of the ranked result. This value is normalized considering an ideal retrieval in which the relevant images appear ahead of any non-relevant one in the ranked result. The relative ordering of the RRSet elements is irrelevant because we are using a binary judgment of relevance. The measure used in QBIC project was defined as $NavgR = A/I$, where A and I are, respectively, the actual average rank and the ideal average rank of the RRSet. In this paper, we propose a slightly different definition of the NavgR measure, inverting the original ratio as shown in Equation 1.

$$NavgR' = \frac{I}{A} = \frac{\sum_{i=1}^{|RRSet|}(i-1)}{\sum_{i=1}^{|RRSet|} rank(i)} \quad (1)$$

In that equation, $rank(i)$ is a function that returns the rank of the i^{th} relevant image. The rank values vary in the range $[0, |DB| - 1]$, where $|DB|$ denotes the cardinality of the image database. With this alternative definition, the NavgR' value ranges from 0 to 1 and equals 1 in the best case, as occurs in precision and recall measures. Higher values of NavgR' are associated to good effectiveness results, instead of vice-versa. This fact reduces the dependence of the measure on the worst results during the process of averaging results from distinct queries.

A drawback with the NavgR' measure, is that, the value obtained for a single query is still very sensitive to the rank of the last relevant document retrieved. In this paper, we deal with this problem considering only a predefined portion of the RRSets for the purpose of NavgR' computation. We consider only the ranks of the first 80% portion of an RRSet. The last 20% of the RRSets (with the highest numerical ranks) are discarded, since any possible misleading judgment of relevance (that will strongly affect the NavgR' value) could materialize itself as a large numerical rank value.

4.1 Experimental Results

Table 2 shows a comparative analysis of the approaches, based on the mean values of NavgR obtained using the results of the 29 processed queries. This table shows relative values, obtained using the GCH approach as reference. We also included the space overhead (in terms of floating-point numbers) and the computational complexity of each approach. Figure 4 on the other hand, shows a summary of the results obtained in terms of Precision vs. Recall curves. Due to the limited space only the curves comparing GCH (typical benchmark), CCV (runner-up among the global approaches), CSH (the best performer among the partitioning approaches) and CBC(3,0.1) (a "good compromise" version of the proposed approach) are shown. A complete evaluation, as well as sample images segmented using CBC, can be found elsewhere [18]. Nevertheless, in general terms, the obtained values of NavgR are consistent with the Precision vs. Recall curves – which further supports the use of NavgR' as a single measure for retrieval effectiveness. Next we discuss the results in terms of global, partition-based and clustering approaches.

Among the global approaches (GCH, CMM and CCV), the traditional GCH has the best performance in both experiments. Clearly, CMM is the worst approach in terms of effectiveness, about 50% worse than GCH's. However, one must note that CMM yields the smallest space overhead, seven times smaller than GCH's overhead. CMM's and CCV's relative performances were also sensitive to the growth, in size and heterogeneity, of the image database.

Among the partition-based approaches (Grid and CSH), CSH is the better approach in both experiments, with a space overhead 50% smaller than Grid's. The effectiveness of CSH is about 38% better than GCH's (at the second experiment of Table 2), at the cost of a space overhead 30 times larger.

Finally, the results in Table 2 show that the three distinct configurations of our clustering approach are more effective than CSH – the best of the compared approaches. The results also show that our approach is more robust to the growth of the database. Considering the $CBC(5, 0.3)$ configuration, we have simultaneously the space overhead of the GCH approach and an effectiveness higher than CSH's effectiveness (at the second experiment), with more robustness since the relative effectiveness of the CSH approach doubled when the database growth while the effectiveness of $CBC(5, 0.3)$ becomes 5 times higher. Using a configuration that results in more regions ($CBC(2, 0.05)$) we can achieve gains of the order of 110% relative to the GCH approach, with more robustness and with a space overhead 50% smaller than CSH's. The robustness is an important factor since it implies that, the larger the database, the larger the differences in effectiveness between our approach and the others compared approaches.

The $CBC(3, 0.1)$ configuration represents an interesting compromise between the size of the metadata (number of regions), effectiveness and robustness. With this configuration, we obtain in average 40 regions per image, resulting in a space overhead 4 times larger than GCH's, but 90% smaller than CSH's. The effectiveness of $CBC(3, 0.1)$ is 98% higher than GCH's while CSH's gain is only of 38%. When the database grows from 1.023 to 20.000 images, the advantage of CSH in relation to GCH becomes approximately two times larger, while with $CBC(3, 0.1)$ this advantage becomes five times larger, suggesting that $CBC(3, 0.1)$ is also considerably more robust than the other compared approaches.

5 Conclusions and Future Work

In this paper, we presented CBC, a new *content-based image retrieval* (CBIR) approach based on cluster analysis. Overall, our contribution, within the ever growing area of

Table 2. Relative gain in effectiveness (using NAvgR') using the GCH approach as reference

Approach	Space*	Complexity	Experiment 1	Experiment 2
GCH	64	$O(n)$	—	—
CMM	9	$O(n)$	-48.44%	-61.00%
CCV	128	$O(n)$	-4.07%	-10.60%
Grid	4096	$O(n)$	9.00%	19.16%
CSH	1856	$O(n)$	16.97%	38.01%
CBC(5, 0.3)	66	$O(n \log n)$	10.56%	53.48%
CBC(3, 0.1)	240	$O(n \log n)$	21.67%	98.09%
CBC(2, 0.05)	930	$O(n \log n)$	33.26%	112.93%

* Average number of floating-point numbers needed to represent each image

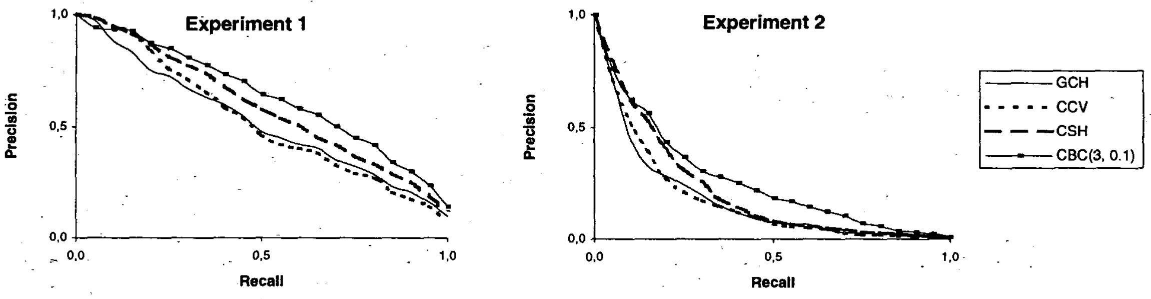

Figure 4. Precision vs. Recall curves for some of the investigated approaches

image databases, is an efficient process to obtain an image's representation (visual metadata) and an effective way to assess similarity between images. To extract an image's color-metadata, we used a simple variation of the single-linkage agglomerative clustering algorithm to find out disjoint regions of the images, composed by pixels with a predefined degree of color similarity. This approach has the advantage of avoiding the notorious problem of color-space quantization, and of being adaptive, in the sense that the obtained segmentation depends mostly on the image itself rather than on "artificial" parameters, such as the number of clusters or the like. Using CBC images are represented and compared based on the set of disjoint connected regions in which they were decomposed – we have also proposed a new distance function to compare two images.

The effectiveness of three configurations of our CBIR approach was compared against the effectiveness of five other approaches, in a controlled environment. Given the results discussed in the paper we believe that the main advantages of our approach were: (1) *flexibility*– using different thresholds for region's size and color's distance, it is possible to obtain different compromises between space overhead and effectiveness; (2) *configurability* – both the clustering algorithm and the distance function can be tuned to work in specific domains, i.e, to consider additionally domain-dependent metadata; (3) *effectiveness* – the three distinct configurations of CBC yielded better retrieval effectiveness than the other five compared approaches; and (4) *robustness* – Our experiments have shown that CBC is more robust with respect to the database growth. In fact, the larger the database size, the larger the relative advantage of CBC. Finally, even though we were not able to perform an object assessesment, it seems that the CBC approach is also effective in terms of processing time. For the sake of illustration, a small sample of images segmented using the CBC approach are shown in the Appendix.

In the near future, we plan to investigate several other aspects related to CBIR systems, such as: (1) region-based queries effectiveness, since our approach has the potential to treat this type of query in a more elegant way; (2) use of additional metadata extraction per region (e.g., shape, texture and topological relations between regions); (3) alternative distance functions; (4) applications of our approach to images of specific domains, where domain-dependent metadata can be explored both during the clustering process as well as when representing/comparing images; (5) a proposal of a retrieval effectiveness measure more adequate than precision, recall and NavgR' to the context of similarity-based queries; (6) design and implementation of an access method to efficiently access images according to their metadata representation and the chosen distance function; and (7) evaluation based on the chosen access method and integration of the efficiency and effectiveness' results in order to compare distinct CBIR approaches.

Acknowledgments

Renato O. Stehling was supported by a Graduate Scholarship from FAPESP, Brazil. Mario A. Nascimento was partially supported by a Research Grant from NSERC, Canada. Alexandre X. Falcão was partially supported by a Research Fellowship Grant from CNPq, Brazil.

References

[1] J. Ashley et al. Automatic and semi-automatic methods for image annotation and retrieval in QBIC. In *Proc. of SPIE – Storage and Retrieval for Image and Video Databases III*, volume 2420, pages 24–35, 1995.

[2] A. D. Bimbo. *Visual Information Retrieval.* Morgan Kaufmann, 1999.

[3] C. Carson, M. Thomas,. S. Belongie, J. M. Hellerstein, and J. Malik. Blobworld: A system for region-based image indexing and retrieval. In *Proc. of the VISUAL'99 Intl. Conf.*, pages 509–516, 1999.

[4] T. H. Cormem, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms.* McGraw-Hill, 1990.

[5] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis.* Wiley-Interscience, 1973.

[6] C. Faloutsos et al. Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, 3(3/4):231–262, 1994.

[7] B. Funt and G. Finlayson. Color constant color indexing. *IEEE PAMI*, 17(5):522–529, 1995.

[8] V. Gaede and O. Guenther. Multidimensional access methods. *ACM Comp. Surveys*, 30(2):123–169, 1998.

[9] R. C. Gonzalez and R. E. Woods. *Digital Image Processing.* Addison-Wesley, 1992.

[10] L. Kaufman and P. J. Rousseuw. *Finding Groups in Data - An Introduction to Cluster Analysis.* Wiley-Interscience, 1990.

[11] H. Lu, B.-C. Ooi, and K.-L. Tan. Efficient image retrieval by color contents. In *Proc. of the 1994 Intl. Conf. on Applications of Databases*, pages 95–108, 1994.

[12] G. Pass, R. Zabih, and J. Miller. Comparing images using color coherence vectors. In *Proc. of the ACM Multimedia'96 Intl. Conf.*, pages 65–73, 1996.

[13] E. J. Pauwels and G. Frederix. Finding regions of interest for content-extraction. In *Proc. of SPIE – Storage and Retrieval for Image and Video Databases VII*, volume 3656, pages 501–510, 1999.

[14] F. Rabitti and P. Savino. An information retrieval approach for image databases. In *Proc. of the 18th Int. Conf. on Very Large Databases*, pages 574–584, 1992.

[15] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, 1983.

[16] S. Sclaroff. Distance to deformable prototypes: Encoding shape categories for efficient search. In *Image Databases and Multi-Media Search.* World Scientific, 1997.

[17] R. O. Stehling, M. A. Nascimento, and A. X. Falcão. On 'shapes' of colors for content-based image retrieval. In *Proc. of the ACM Intl. Workshop on Multimedia Information Retrieval - MIR'00*, pages 171–174, 2000.

[18] R. O. Stehling, M. A. Nascimento, and A. X. Falcão. An adaptive and efficient clustering-based approach for content based retrieval in image databases. Technical Report 01-03, Dept. of Computing Sciences, Univ. of Alberta, March 2001.

[19] M. Stricker and M. Orengo. Similarity of color images. In *Proc. of SPIE – Storage and Retrieval for Image and Video Databases III*, volume 2420, pages 381–392, 1995.

[20] M. J. Swain and D. H. Ballard. Color indexing. *Intl. J. Computer Vision*, 7(1):11–32, 1991.

Appendix

Sample images segmented using the CBC approach.