Efficient Content-based Image Retrieval through Metric Histograms

A. J. M. TRAINA, C. TRAINA JR., J. M. BUENO and F. J. T. CHINO ([agma, caetano, josiane, chino]@icmc.usp.br)
Computer Science Department - University of Sao Paulo at Sao Carlos - Brazil

P. AZEVEDO-MARQUES (pmarques@fmrp.usp.br)
Science of Image and Medical Physics Center Medical School of Ribeirao Preto - University of Sao Paulo at Ribeirao Preto - Brazil

Abstract. This paper presents a new and efficient method for content-based image retrieval employing the color distribution of images. This new method, called *metric histogram*, takes advantage of the correlation among adjacent bins of histograms, reducing the dimensionality of the feature vectors extracted from images, leading to faster and more flexible indexing and retrieval processes. The proposed technique works on each image independently from the others in the dataset, therefore there is no pre-defined number of color regions in the resulting histogram. Thus, it is not possible to use traditional comparison algorithms such as Euclidean or Manhattan distances. To allow the comparison of images through the new feature vectors given by metric histograms, a new metric distance function *MHD*( ) is also proposed. This paper shows the improvements in timing and retrieval discrimination obtained using metric histograms over traditional ones, even when using images with different spatial resolution or thumbnails. The experimental evaluation of the new method, for answering similarity queries over two representative image databases, shows that the metric histograms surpass the retrieval ability of traditional histograms because they are invariant on geometrical and brightness image transformations, and answer the queries up to 10 times faster than the traditional ones.

Keywords: Color histograms, Content-based image retrieval, CBIR, Image similarity retrieval, Image features, Image indexing.

1. Introduction

The importance of the World Wide Web (WWW) as a data repository is unquestionable due to its increasing and ubiquitous use as a means for fast access to information as well as its dissemination. Taking advantage of theses benefits given by the WWW, more and more computer applications have been developed to run over the internet. Examples of such applications include digital libraries, and information systems for education and business, among others. Therefore, the amount of data available and searched for in the internet is growing tremendously, with a perspective of even faster increasing in the future. Within this scenario, it is clear that such systems will demand greater efforts for better data organization, classification and retrieval.

Other important point about the data available over the internet is their variety and lack of structure. The data comes from simple texts and numbers to audio, images and video recordings. Consequently, to deal with data from the WWW it is frequently necessary to employ the technology developed for processing multimedia data.

Because of the overwhelming increase in the volume of multimedia data, it is necessary to use more efficient and sophisticated methods to organize such data. The Database Management Systems (DBMS) are used since the 60's to guarantee the integrity of simple data (numbers and short texts), allowing fast and effective searching and retrieval of them. The main practice exploited to accelerate data retrieval operations in database management systems is indexing the data through an access method tailored to the domain of the data. The most well-known access methods for simple data are the B-tree and its successor B+-tree. The data types stored in traditional databases (numbers and small character strings) using B-trees exhibit the property of total ordering. That is, it is clear that the number 115 is less than 235, and the string "*house*" is lexicographically anterior to the string "*hunter*". Also, it is possible to establish a proper order among every element of such data types. B-trees use this property to prune large quantity of comparison operations maintaining the data properly sorted. However, when a data type do not have this property, as happens with images, video and all the so-called multimedia information, the traditional access methods cannot be used anymore. For example, given two images, it is not possible to compare both and say which one is "less than" or "greater than" the other. Moreover, multimedia data are embedded in multi-dimensional or even non-dimensional domains which does not directly allow sorting. In this paper we focus on images, which are largely present on the Web and are also frequently used during searching sessions.

Multi-dimensional data can be indexed through spatial access methods (SAM), which have been developed for spatial or multi-dimensional domains. The main representative of this type of access method is the well-known R-tree [28].

The process of searching over large volumes of images is an arduous task. The common approach is the sequential search over the full set of images. However, this approach becomes impractical when the database size is large, owing to both the IO cost of retrieving each image, and the high computational cost of comparing two images, multiplied by the number of images pertaining to the image set. During the sequential scanning, every image of the set must be accessed and compared with the reference image given in the query by the user. Thus, as the size of the image set grows, the time required to get the answer increases as well.

There are two main approaches to search for images in an image database. The first one attaches textual description to each image, and the search process finds the required images through these texts. This is not a good solution as, besides it cannot be automated, the retrieval depends on the objective of the operator when he or she was describing the image, not on the objective when the query is issued. The second approach uses a set of image processing algorithms to automatically extract characteristics (features) from the images, and those characteristics are then indexed together with the image. In this approach, the image retrieval operation processes the image features in place of the images. This approach is independent of human subjectivity, and is the one adopted in this paper.

The indexing of the extracted characteristics is usually performed through spatial access methods, where each extracted feature is deemed as a dimension. For example, if the properties analyzed are the average brightness level of the image together with its standard deviation, one can say that the feature vector has two dimensions, one for the average and other for the standard deviation.

Image histograms can be seen as multi-dimensional vectors, where each bin, represented by a vector entry, is seen as a dimension of the data. Therefore, an image histogram of an image quantized in 128 colors or gray levels will have a histogram represented by a vector with 128 elements or bins. Therefore, for image histograms, each bin is indexed as a dimension. However, the spatial access methods are efficient up to a rather limited number of dimensions, in general not larger than 10 or 15 [15]. Therefore, indexing images through their histograms (usually with 256 bins or more) is highly inefficient. It has been shown [33] that the main factor affecting the efficiency of a multi-dimensional access method is the intrinsic dimensionality of the dataset, that is, the number of uncorrelated dimensions. Therefore, indexing operations could be done more efficiently if the correlations between close bins could be used. Some attempts have been done before to find more concise representations of histograms using dimensionality reduction techniques [12] [9] [34], but all of them rely on a predefined number of dimensions. Therefore, none benefit from existing correlations among close bins at each image, and the particularities of each individual image are not taken in account.

In this paper we consider that histograms of two images may have different correlation between its bins, so reduced histograms from distinct images could have distinct number of dimensions. This points out that these reduced histograms are in a domain that is not spatial anymore - this domain does not have a fixed number of dimensions defined, as each reduced histogram has a different number of “reduced

bins", and therefore they cannot be indexed through any spatial access method. However, if a proper distance function is defined, these reduced histograms can be seen as being in a metric domain, so we call these the "metric histograms". We will show in this paper that by using such reduced histograms, the efficiency and the efficacy of image retrieval by content operations are much better than using traditional histograms. Experiments showed that searching operations using metric histograms can be up to 10 times faster than using traditional histograms, even when using the same access method.

The remaining of this paper is structured as follows. In the next section, we state the problem tackled by the new content-based image retrieval technique and its applicability over the internet. Section 3 presents the main concepts necessary to understand the scenarios where the contributions of this paper are applicable. Section 4 introduces the metric histograms as well as the $MHD(\ )$ distance function to be used to measure the dissimilarity between images represented by their metric histograms. Section 5 presents the results obtained by the experiments performed over metric histograms in order to evaluate the usefulness of them, regarding retrieval accuracy, time measurements, and invariance to spatial resolution when the images are given by their thumbnails. Section 6 discusses related work, and section 7 presents the conclusion of this paper. A shorter version of this paper was published in the IFIP Sixth Working Conference on Visual Database Systems [52].

2. Problem Definition and Applicability

The work presented in this paper intends to support the development of an efficient technique for image retrieval by content to support medical applications distributed over the web. This technique should work well on images in normal size as well as on images with reduced size, or thumbnails, because over the WWW it is common to reduce the data size in order to gain performance. Thumbnails are images presented in low spatial resolution, and they are frequently used when communicating images over the internet during a process of selection guided by the user.

The technique shown herein is part of the development of a picture archiving and communication systems (PACS) [48] with content-based image retrieval facilities, which can run over the internet.

Initially, PACS were introduced as image management systems aiming at improving the efficiency of radiology practice, making the information generated by tomographs and other image-generation devices of medical exams (ultrasound, direct radiology among others)

broadly available through the use of network protocols. The effort of keeping together all the information concerning the patients has evolved into PACS integrated with the Hospital Information System and with Radiology Information Systems (HIS and RIS), allowing one to keep together data in many formats, including texts, numbers, images, audio and video. Figure 1 shows a general architecture of a PACS, where the database server, as well as the image server and HIS/RIS servers are connected through an intranet to the image acquisition center and to the diagnosis center. The users' terminals are connected through this intranet by a firewall. A long distance connection through the World Wide Web will only access the PACS through the firewall as well.

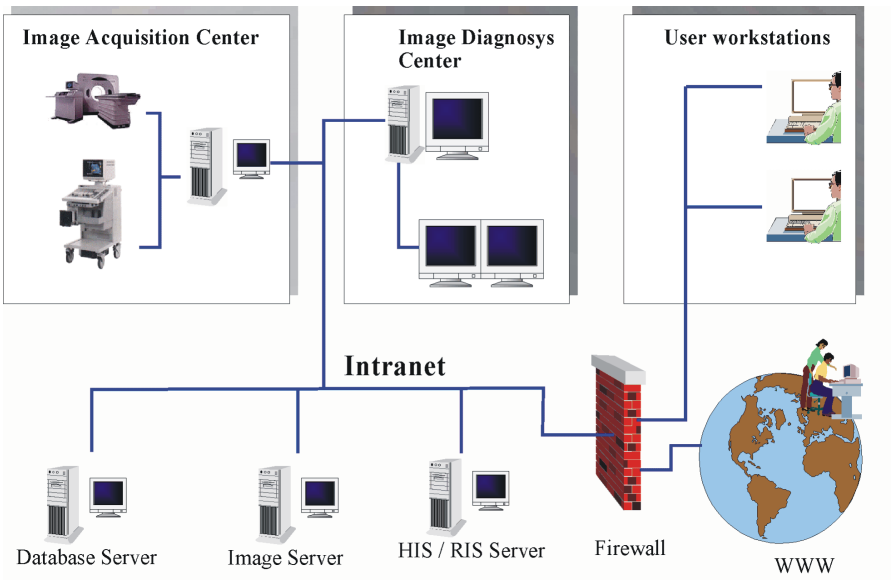

Figure 1. A general architecture of a Picture Archiving and Communication System (PACS) allowing its access by the World Wide Web.

Traditional PACS integrated with Hospital Information Systems (HIS) allow physicians to ask questions like that:

1 - "*Find the patients admitted since 01-Aug-1999 who presented coronary occlusion*". However, if the physician is interested in to retrieve the most similar chest MRI[1] of the patient John Doe, that is:

2 - "*Find the most similar image to the chest image of patient John Doe*". Question 2 would not be answered by any conventional PACS or Radiology Information System. This is due to the fact that "from the shelf" DBMS cannot compare images as they do with dates, numbers or short texts. Therefore, they cannot answer the question 1 stated

[1] MRI stands for magnetic resonance imaging.

above. Ideally, the images of medical exams should be kept along with conventional data (text and numbers). Therefore, it would be possible to ask both types of queries regarding the content of images and the usual ones based on textual information.

Images already stored in a PACS could help improving the medical diagnosis even more if they could be compared and searched for by their content, allowing the same type of comparisons that can be done on alphanumerical information. A PACS that allows images to be searched by content would be an effective and valuable tool to assist the physicians in issuing diagnosis [47].

As the human beings have much more ability to remember information presented as graphics and images, it is worth to develop a system that helps a physician to "remember" or "recollect" cases previously diagnosed and treated. This tool can be valuable to help and enhance medical diagnosis too. However, to compare and deal with images as a simple data type, it is necessary to develop techniques, so software systems can treat them equivalently to the way that human beings do. Such techniques should be able to convey the "essence" of the images, allowing automatic image comparison, getting the most similar ones from a data base. This paper addresses this problem, bringing in a novel technique for content-based image retrieval (CBIR) using the brightness distribution of images represented by their histograms. Moreover, such histograms take advantage of the existing correlation among adjacent histogram bins. This is done in a way that allows comparisons between images with different sizes, placements and brightness transformations.

3. Main Concepts

As the volume of images grow, the image database management systems need to rely on indexing or access methods to store and retrieve traditional data (texts, numbers and dates) as well as image data. The design of efficient access methods has attracted and motivated researchers of the database field for more than three decades. Thus, there are many consolidated access methods for traditional data, such as the B-tree [18] and hash-based indexing access methods [24].

Spatial data, such as regions of maps given by their multidimensional coordinates, have been managed by the so called Spatial Access Methods - SAMs, which include treelike structures, such as the R-tree [28], R+-tree [45], R*-tree [7], k-d-B-Tree [42], space- filling curves [30], spatial hash files [43], etc. An excellent survey on SAMs is given in [22]. Spatial domains have also been widely used to store feature sets

extracted from images and other complex data. Therefore, whenever the features extracted have a fixed number of attributes a SAM can be applied to index such data

The majority of features extracted from images can be seen as multidimensional points in a n-dimensional space. This is the case of histograms, moment invariants, Fourier features, wavelet coefficients, principal component analysis values, among others. The problem with the spatial access methods is that they downgrade the indexing and the retrieval of objects as the dimension of the objects increases. Thus a sequential scan over the whole dataset would outperform many of such methods. However, other access methods can deal with high-dimensional datasets and can be used to answer queries on such high-dimensional objects. This is the case of the X-tree [8], TV-tree [36] and the Hybrid-tree [15], which were developed aiming to manage high dimensional data. However, they already require that all feature vectors have the same number of components (dimensions). If this requisite cannot be met, it is not possible to use a SAM at all, and other types of access methods should be used. An approach for overcoming this drawback is through the Metric Access Methods, which will be discussed next.

3.1. Similarity Queries

The Metric Access Methods (MAMs) are more general than SAMs and can deal with data with varying number of dimensions. The only information available in metric domains are the objects and the distances between them, because there is no spatial placement or relationship among the objects. A proper distance function is usually provided by a domain expert, who gathers the main properties and specific particularities of the target domain in order to compare the data. Also, MAMs were first developed aiming at supporting the processing of similarity queries [16]. The following definitions will help us to organize the concepts concerning the approach taken in this paper, and Table I summarizes the main symbols used in this paper.

Definition 1 (*Metric distance function*): Given three objects, s_i, s_j and s_k pertaining to the domain of objects $\mathbb{S}$, a distance function $d(\ )$ is said to be metric if it satisfies the following three properties:

symmetry: $d(s_i, s_j) = d(s_i, s_i)$,
non-negativity: $0 < d(s_i, s_j) < \infty, i \neq j$ and $d(s_i, s_i) = 0$,
triangle inequality: $d(s_i, s_j) \leq d(s_i, s_k) + d(s_k, s_j)$.

Table I. Summary of symbols.

Symbols	Definitions
$\mathbb{S}$	domain of objects
S	set of objects in domain $\mathbb{S}$
s_i, s_j, s_k	objects in domain $\mathbb{S}$
$d()$	distance function
s_q	a query object (or query center)
r_q	radius of a range query
k	number of neighbors in an NN query
$M_H(A)$	metric histogram of image A
N_A	number of buckets in the metric histogram of image A
$< b_k, h_k >$	definition of bucket k in a metric histogram, where b_k is the index of the rightmost bin of the original histogram represented in bucket k, and h_k is the normalized value of the rightmost bin represented in bucket k
MAM	metric access method
SAM	spatial access method
PACS	picture archiving and communication system

A metric distance function is the foundation to build MAMs, which were developed since the pioneering work of Burkhard and Keller [14] and have recently achieved an efficiency level good enough to be used in practical situations. This is obtained using the triangle inequality property to prune blocks of objects, as the B-trees do using the ordering property of ordered domains. This is the case of the M-tree [17], the Slim-tree [54] and the Omni-family members [44].

Definition 2 (*Metric Space*): Given a set of objects $S \subset \mathbb{S}$ pertaining to a given domain and a metric distance function $d()$ for these objects, $\mathbb{M} = (\mathbb{S}, d)$ is called a Metric Space.

Spatial datasets following an L_p-metric (such as Euclidean distance) are special cases of metric spaces, because it is possible to state the distance between any two objects of the dataset. To support image retrieval based on the similarity of the image content, it is first necessary to define what *similarity* mean. Actually, a distance function $d()$, as stated in Definition 1, measures the dissimilarity between images. When the value of $d()$ is equal to zero the images are equal or equivalent under the criterium used by the distance function. The higher the value of the distance, the more different two images are. In this paper, we considered two classes of similarity queries: range queries and nearest-neighbors queries, which are defined as follows.

Definition 3 (*Nearest-Neighbor query*): Given a query object $s_q \in \mathbb{S}$ (the object domain), the nearest-neighbor is the unitary subset of S such that $NNquery(s_q) = \{s_n \in S | \forall s_i \in S : d(s_n, s_q) \leq d(s_i, s_q)\}$.

The dataset is searched for and the nearest object from the query object s_q is provided. A common variation is the k-nearest-neighbor query, which finds the k closest objects to s_q in the dataset. A typical example of a k-nearest-neighbor query with $k = 5$ in an image database domain is: "*find the 5 nearest images to image A*".

Definition 4 (*Range query*): Given a query object $s_q \in \mathbb{S}$ and a maximum search distance r_q, the answer is the subset of S such that $Rquery(s_q, r_q) = \{s_i | s_i S : d(s_i, s_q) \leq r_q\}$.

The dataset is scanned and all the objects whose distance to the query object center s_q is less or equal to the query radius r_q is retrieved. An example of a range query is: "*find all the images that are within 10 units of distance from image A*", or "*find the images in the database that are at least 80% similar to the given image A*", where *image A* is the query object.

3.2. Image Features

Images can be described by their main properties or characteristics. The properties are usually grouped in arrays, and give the "essence" of the images by following a specific criterium. Thus, we have the following definition:

Definition 5 (*Feature Vector*): Given a specific criterium or image property, the feature vector is a concise representation of an image.

It is well-known that the properties of color, shape and texture are the main image characteristics or properties of images [5] [56] [49] [26]. Therefore, such characteristics of features would be used in comparisons and content-based image retrieval (CBIR). More than one criterium can be used to generate a feature vector. For instance, one could extract characteristics of color and shape and build a feature vector regarding these two criteria for the image database. Afterwards, it would be possible to perform comparisons over images regarding these two aspects.

The comparison of images is done by calculating the distances among them, which usually are costly operations, as images are complex objects. Hence, minimizing these calculations is one of the most important aspects to obtain an efficient answer for the queries. The metric access

methods (MAM) were developed to minimize the number of distance calculations taking advantage of the triangular inequality property of metric distance functions (see Definition 1), which allows to prune parts of the tree during the answering process. Metric access methods built on the image features have been successfully used to index images and are suited to answer similarity queries [10] [13].

Besides the use of access methods to index the features of images and accelerate the retrieval of images, many techniques for image comparison, using color histograms [32] [40] [41], shape [23] [19] [55] and texture [58] [25] [51] have been proposed. The algorithms to extract shapes and textures are very expensive and dependent on the application domain, so it is better to leave them as the last step for separating the images, when the candidate response set was already reduced using the other features. The importance of color or brightness histograms is due to the simplicity of getting and comparing them, operations that are performed in linear time.

Note that a system for content-based image retrieval works on comparing images through the comparison of their feature vectors. Therefore, the extraction of image features should be done as automatically as possible. Also, the features should discriminate the images independently of transformations (scale, translation, rotation and brightness). That is, the features should be invariant to image transformations. Section 4 presents the main contribution of this paper, the metric histograms, which are invariant to all aforementioned transformations.

3.3. How to Measure the Effectiveness of Retrieval Processes

A human specialist on the data domain can easily verify if the CBIR techniques employed to answer similarity queries have produced satisfactory results, when selecting the most similar images to a given image query. However, it is necessary to provide an objective and quantitative method to measure the retrieval effectiveness of a new method. It is important to validate a method and state its real performance, allowing comparison with other retrieval techniques. The *Precision* and *Recall* approach have been widely used in this regard [12] [57] [1] [6]. *Recall* indicates the proportion of relevant images in the database which has been retrieved when answering a query. *Precision*, by the other hand, is the proportion of the retrieved images that are relevant for the query. Therefore, precision gives a direct value for measuring the retrieval ability of the method. The usefulness of the retrieval method is proportional to the value obtained, that is, values next to zero mean poor

mapping to the user requirements, and values next to one means high mapping to the user specifications.

Formally, let X be the set of relevant images for a given query, Y the set of retrieved images and x, y and z be respectively the number of images in the sets $X \cap Y$, $\bar{X} \cap Y$ and $X \cap \bar{Y}$. Thus, precision and recall can be expressed through the following conditional probabilities:

$$precision = P(X|Y) = \frac{P(X \cap Y)}{P(Y)} = \frac{x}{x+y}$$

$$recall = P(Y|X) = \frac{P(Y \cap X)}{P(X)} = \frac{x}{x+z}$$

Ideally the set of objects retrieved should be equal to the set of objects relevant for a specific query. However, due mainly to the limitations on working with the features extracted from images as well as the distance function used to compare them, this would not happen as the number of retrieved objects grows. Through the analysis of the graphs of precision and recall for increasing number of objects retrieved by the queries over the dataset, we can sense the behavior of the retrieval technique to bring the correct answer for the given query. A common measurement is the area under the graph, where the larger the value obtained the better. However, the area alone does not give the idea of the behavior of the method regarding the local performance. Thus, the better way of learning "how the read" a precision-versus-recall graph is by example, as we will show in section 5.

4. The Metric Histograms

In this section we present a new CBIR approach based on brightness distribution of images. This new approach is called Metric Histograms, and works taking advantage on correlations among adjacent bins of histograms. This will lead to a more compact histogram representation and have also the desired property of being invariant to image transformations, as we will show next.

The usual representation of images is a set of elements (pixels) placed on a regular grid. The pixel values are obtained from a quantization process and correspond to the brightness levels, ranging from zero to the maximum value given by the pixel depth stated. Thus, formally an image can be represented by the following notation:

Definition 6 (*Image*): An image A is a function defined over a two-dimensional range $G = [0, x_0] \times [0, y_0]$ taking values on the set of possible brightness values $V = [0, v_0]$. That is, $A = \{(x, y, v(x, y))/(x, y) \in G \text{ and } v \in V\}$.

Definition 7 (*Histogram*): The histogram $H(A)$ of an image A provides the frequency of each brightness value z in the image. The histogram of an image with t brightness levels is represented by an array with t elements, called bins.

Definition 8 (*Normalized Histogram*): The normalized histogram $N_H(A)$ of an image A provides the frequency of each brightness value z in the image, given in percentage.

The normalized histogram of an image with t brightness levels is also represented by an array with t elements.

From now on in this paper, whenever we refer to a histogram, we mean a normalized one. To obtain the normalized histogram of an image is not a costly operation. The normalized histogram is invariant to geometrical transformations. That is, normalized histograms allow comparisons between images of any size, so geometric transformations performed on the source images will give the same histogram. Figure 2 presents an image obtained from magnetic resonance tomograph and its associated normalized histogram. This image has a spatial resolution of 512x512 pixels displayed in 256 brightness levels, thus its histogram has 256 bins. Indexing histograms like this one requires indexing arrays with 256 elements or, in terms of access methods, dimensions. The distance (or dissimilarity) between two histograms can be measured as the summation of the difference between each bin, which is known as the Manhattan distance function or L_1-norm. The histogram is usually taken as a feature vector representing the global brightness distribution of the image.

In this work, we assume that the brightness level representations are similar to its adjacent representation levels, so the shape of the histogram can be kept using an approximation of it. Therefore, we propose to represent an approximation of a normalized histogram through a set of line segments. That is, a piecewise approximation is done over the histogram. As each image histogram presents a particular shape and correlation among neighbor bins, these histograms can be approximated by a different number of lines or pieces. So the histogram approximation by linear pieces can be optimized to describe each one of them. As there is no predefined number of pieces to build the histogram approximation, and if there is a metric distance function defined for the objects (*feature vectors*) given by this new representation, then this new histogram representation is in a metric domain. We will show in this section that this is true. The next definitions state formally the points just discussed above.

Definition 9 (*Metric Histogram*): A Metric histogram $M_H(A)$ of an image A is defined as $M_H(A) = \{N_A, < b_k, h_k > | 0 < k <$

$N_A\}$, which is a set of N_A buckets b_k, each one with the normalized height h_k.

A normalized histogram is composed by a number of bins. This number depends on the brightness resolution of the image, so it is a fixed number. In a metric histogram, the equivalent to the histogram bin is called a *bucket*. Each bucket corresponds to a linear segment, or line for short, in the approximation. Buckets do not need to be regularly spaced. The number N_A of buckets in a metric histogram depends on the acceptable error in the approximation process and on the image itself. Each bucket k is a pair $< b_k, h_k >$, where b_k is the index of the rightmost bin of the original histogram represented in bucket k, and h_k is the normalized value of the rightmost bin represented in bucket k. Notice that b_0 is always zero. To simplify the notation, we indicate the bucket b_k of the metric histogram of image A as A_{bk}, and the normalized value h_k of the metric histogram of image A as A_{hk}. Figure 4(a) presents the metric histogram of two images A and B, and the components of their metric histograms.

The algorithm used to generate the metric histogram finds the *control points*, which are the inflection, minimum and maximum local points on the histogram, in order to create the approximation curve on the normalized histogram. In this way, the histograms are seen as

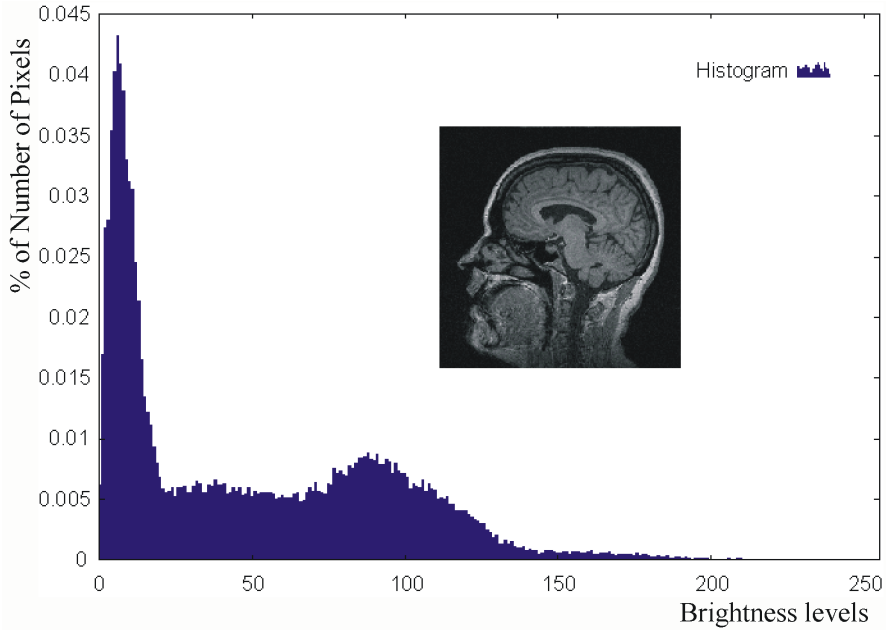

Figure 2. An MR image and its normalized histogram.

functions in a 2-D space. Figure 3 shows a metric histogram obtained from a normalized histogram of the same image shown in figure 2.

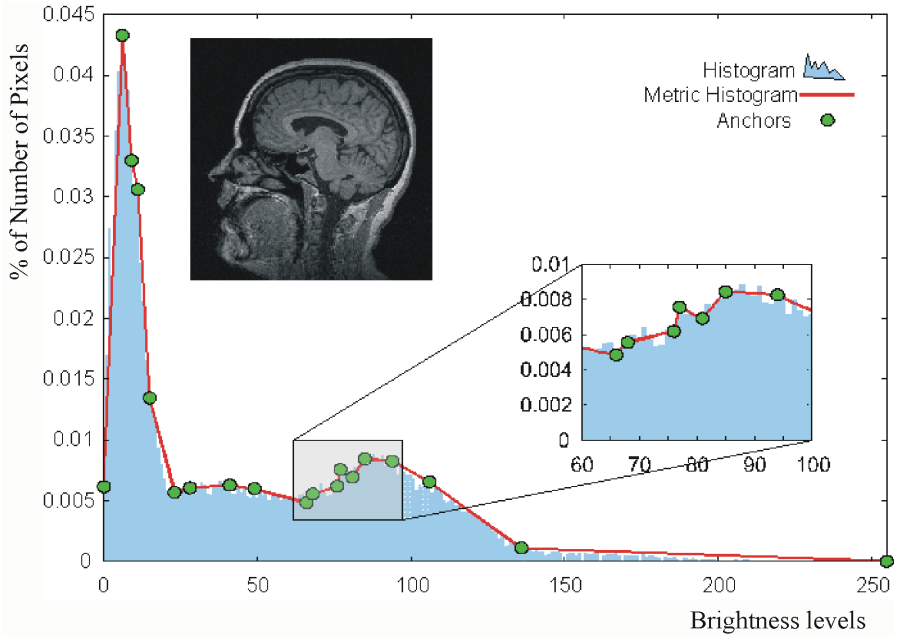

Figure 3. Deriving a metric histogram through the piecewise approximation over the normalized histogram.

Regarding metric histograms, a question that arises is how to compare them, as the number of buckets and the spawning of the buckets from different histograms are variable. That is, the usual distances for histograms, such as Euclidean, χ-square, Manhattan, Kolmogorov-Smirnov, Kuiper, between others [12] cannot be used. Therefore, we developed a new metric distance function in order to quantify the dissimilarity between metric histograms.

Definition 10 (*Metric Histogram Distance* - MHD): The distance function $MHD()$ between two metric histograms is given by the non-overlapping area among the two curves representing the metric histograms, i.e.

$$MHD(M_H(A), M_H(B)) = \int_{x=0}^{steps} |M_H(A,x) - M_H(B,x)|dx$$

Figure 4 gives a graphical example of how to calculate the distance between two metric histograms. Figure 5 depicts the algorithm developed to calculate this distance function. In figure 4(a) the two histograms are overlapped, and the intersecting points and the ones that limit the

buckets are shown. Figures 4(b) to 4(d) show how such points are used to calculate the area inside each region (in the steps of the algorithm in Figure 4). Note that the number of steps is greater than or equal to the number of buckets of the histogram with more buckets. This is due to the fact that as the width of the buckets is variable, in some occasions they must be divided in order to obtain the area between the two metric histograms more easily. This is exemplified in Figure 4(b). Figures 4(c) and 4(d) shows the next two steps performed to calculate the $MHD()$ function, helping to understand how the algorithm works (five more steps will be needed to finish to compute the distance among them). When one of the metric histograms finishes before the other one, that is, when $N_A < N_B$, the calculation of the distance also stops, considering that the finished metric histogram goes to zero.

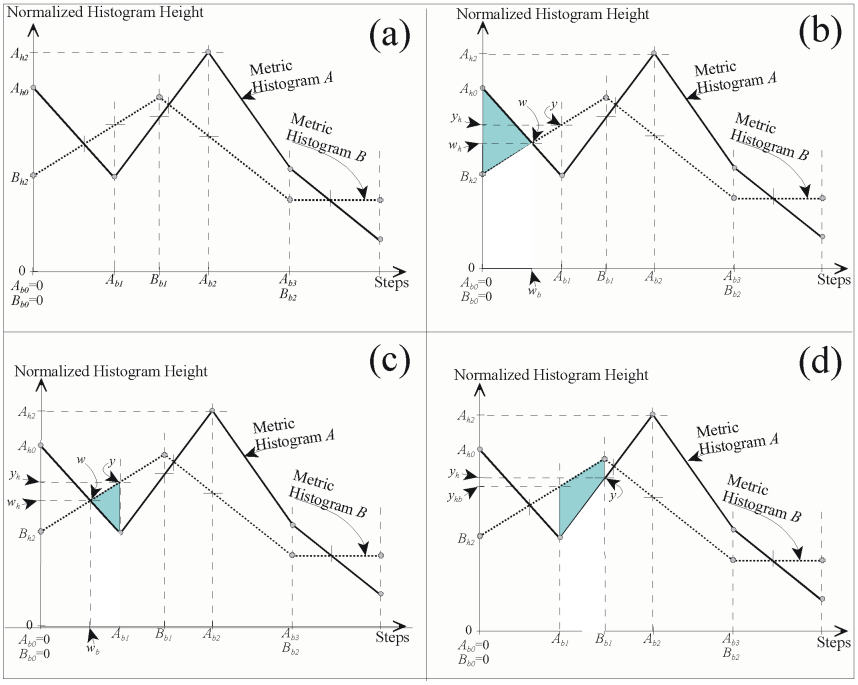

Figure 4. Calculating the distance between two metric histograms by computing the area between the curves. (a) Two metric histograms A and B, and the points used to specify the steps of the algorithm. (b) First step of the algorithm, exemplifying when the two M_H intersects each other. (c) Second step of the algorithm. (d) Third step of the algorithm.

It is important to show that $MHD()$ complies with the three axioms of metric distance functions. The distance $MHD()$ is calculated as a summation over all steps (a bucket or a division of it, if there is intersection between the metric histograms). Thus, by its definition,

Algorithm 1 Algorithm for calculating the MHD distance function.

The MHD distance function: calculating the distance between two metric histograms

Require: the two metric histograms $M_H(A)$ and $M_H(B)$
Ensure: the distance between $M_H(A)$ and $M_H(B)$

set $dist = 0$, $s = 0$, $a = 0$, $h_a = A_{b0}$, $h_b = B_{b0}$, $i = 1$ and $j = 1$
while there is more steps to compare **do**
  if bucket $A_{bi} < B_{bj}$ **then**
    calculate the value of B at position A_{bi} as $y = (A_{bi}, y_2)$
    set $b_m = A_{bi}$, $base = bm - s$, and $y_1 = A_{hi}$
    increment i
  else
    calculate the value of A at position B_{bj} as $y = (B_{bj}, y_1)$
    set $b_m = B_{bj}$, $base = bm - s$, and $y_2 = B_{hj}$
    increment j
  end if
  if line $((a, h_a), (a, h_b))$ intersects line $((b_m, y_1), (b_m, y_2))$ **then**
    calculate the intersection $w = (w_b, w_h)$
    calculate

$$area_1 = \left|(w_b - a) * \frac{h_a - h_b}{2}\right|$$

and

$$area_2 = \left|(b_m - w_b) * \frac{y_1 - y_2}{2}\right|$$

    add $area_1 + area_2$ to $dist$
  else
    calculate $area_1 = base * \frac{h_a + y_1}{2}$ and $area_2 = base * \frac{h_b + y_2}{2}$ and
    add $|area_1 - area_2|$ to $dist$
  end if
  set $h_a = y_1$ and $h_b = y_2$
  set $a = base + a$
end while
return $dist$

Figure 5. Algorithm for calculating the MHD distance function.

the distance $MHD()$ is clearly symmetric and non-negative. Without loss of generality we can assume that the triangular inequality holds for a step. Since the $MHD()$ distance is the summation of the difference of non-overlapping areas between a pair of parallelograms, the triangular inequality also holds for every step of the $MHD()$ calculation. Thus, by induction over the steps, the triangular inequality also holds for the

whole distance computation. More formally, we have the proofs for all the three axioms:

– **Symmetry**: Given two metric histogram $M_H(A), M_H(B)$, the distance between them is calculated by:

$$\begin{aligned} MHD(M_H(A), M_H(B)) &= \int_x |M_H(A,x) - M_H(B,x)|dx \\ &= \int_x |M_H(B,x) - M_H(A,x)|dx \\ &= MHD(M_H(B), M_H(A)) \end{aligned}$$

■

– **Non-negativity**: The $MHD()$ distance is by definition a summation of absolute values, so cannot result in a negative value. It will only be equal to zero when calculated on itself.

$$MHD(M_H(A), M_H(A)) = \int_x |M_H(A,x) - M_H(A,x)|dx = 0$$

■

– **Triangle inequality**: Let $M_H(C)$ be a third metric histogram. Then we have

$$MHD(M_H(A), M_H(C)) = \int_x |M_H(A,x) - M_H(C,x)|dx$$

By the non-negativity axiom, we can add the distance between $M_H(B)$ to itself, because it is equal to zero:

$$\begin{aligned} &MHD(M_H(A), M_H(C)) \Rightarrow \\ &\int_x |M_H(A,x) - M_H(C,x)|dx + \int_x |M_H(B,x) - M_H(B,x)|dx \\ &= \int_x |M_H(A,x) - M_H(C,x) + M_H(B,x) - M_H(B,x)|dx \\ &= \int_x |M_H(A,x) - M_H(B,x)|dx + \int_x |M_H(B,x) - M_H(C,x)|dx \\ &= MHD(M_H(A), M_H(B)) + MHD(M_H(B), M_H(C)) \end{aligned}$$

■

Our experiments show that the number of buckets in the metric histogram is much smaller than the number of bins of normalized histograms, coming from 256 bins to figures around 12 to 32 buckets in the metric histogram. It is important to recall that the metric histograms are obtained from normalized histograms. Thus, the following properties hold:

Property 1 - An original image and the same one scaled, translated or rotated will have the same metric histogram.

Property 2 - The metric histograms are curves in the space, thus the metric histograms can be adjusted at its beginning and ending. Therefore, metric histograms are also invariant to the linear brightness transformations on images.

These two properties highlight that some restrictions of using histograms for image retrieval are overcame through the use of metric histograms. That is, the metric histograms become invariant to geometric transformations (scale included) and linear brightness transformations.

5. Experimental Results

In this section, we show and discuss some experimental results aiming to evaluate the performance of our proposed techniques. The proposed metric histogram and the distance $MHD()$ have been implemented and used in a PACS that embodies content-based image retrieval capabilities, under development in the University of Sao Paulo [4].

We have used two image databases in the experiments. The first one, named "*MRHead500*", has 500 human brain images obtained by magnetic resonance tomography (*MRT*), and the second one, named "*MRVarious4247*", has 4,247 images from various human body parts also obtained by *MRT*. Each image is originally in the *DICOM* format with 256 brightness levels and different spatial resolution. As the objective was to evaluate the performance and usefulness of such new methods for answering similarity queries, we employed the Slim-tree as the metric access method to perform such queries. The Slim-tree has been used because it allows the minimization of the overlap between nodes in the structure as well as to measure this overlap. This overlap minimization leaded to big improvement on performance when answering similarity queries [53]. So far, only the Slim-tree has such quality, which has been shown to be useful when dealing with multi-dimensional and non-dimensional datasets, as is the case of normalized and metric histograms.

The experiments performed aimed to test how the proposed method works on different aspects of image retrieval by content. The description of the tests and the results obtained are discussed as follows. We have conducted two sets of experiments. The first one was performed aiming to verify the properties of the metric histograms, as well as comparing the new dissimilarity function $MHD()$ used in image comparisons and indexing operations. The second set of experiments targets to evaluate the use of metric histograms as a way to compare images, regardless their spatial resolution, and evaluate the use of the function $MHD()$ as a suitable dissimilarity function to perform searching operations over the WWW.

5.1. Comparing the Metric Histogram with the Normalized Histogram

The first set of experiments intended to measure the power of image discrimination given by the metric histogram as compared to the normalized histograms over the same set of images. The objective of these experiments was to test whether the metric histogram should be used in place of the normalized histograms on indexing operations, comparing the precision and recall properties of metric histograms, and the speed up obtained when using the metric histogram.

For these experiments we extracted the normalized histogram and generated the metric histogram of each image, creating two sets of features for each database. Each set of features was indexed using the Slim-tree structure, so the similarity queries were processed by this access method. Therefore, four Slim-trees were built in total for this experiment:

- a tree for the "*MRHead500*" dataset using the normalized histogram,
- a tree for the "*MRHead500*" dataset using the metric histogram,
- a tree for the "*MRVarious4247*" dataset using the normalized histogram, and
- a tree for the "*MRVarious4247*" dataset using the metric histogram.

Regarding the image comparisons using the normalized histograms, they were compared using the Manhattan distance function applied over the 256-element feature vector. The metric histograms were compared using the $MHD()$ distance function proposed, working on the number of buckets varying from 11 to 32. It is important to emphasize

that the Manhattan distance function is the least expensive distance measurement of the L_p norm family. Therefore, we compared the new $MHD()$ distance function with the most efficient and simplest method from the literature.

5.1.1. *How Well Metric Histograms Discriminate Images*
In this set of experiments, we intended to verify whether metric histograms are able to be used in place of normalized histograms when performing image comparison and indexing. To answer this question, we considered that the results provided by queries executed using the normalized histogram, which is the traditional approach, are the corrected ones, and compared these results with those provided by queries executed using the metric histogram. The objective was to compare the images retrieved through their metric histogram with the images retrieved through their normalized histogram.

It is important to remember that the comparison results obtained by any compressed representation of data, as are both kind of histograms, are not completely correct anyway, mainly because in this case the "correct" answer involves more subjective matters, which are difficult to represent by only a type of feature vector. It was interesting to perceive that the obtained results for a similarity query through the use of the metric histogram can be considered better than the results obtained by answering the same query through normalized histograms. This effectively occurred when we submitted pairs of results to physicians. A result of applying an 8-NN query on the image 287 from the dataset "*MRHead500*" is presented in Figure 6. As we can see, the sequence of decreasing similarity makes more sense when answering the query through metric histograms than when evaluating the query through the normalized histogram.

To evaluate if the metric histograms are suitable replacements to the normalized histograms, we have compared their figures of precision and recall as explained in section 3. Figures 7 and 8 present the *precision* and *recall* curves obtained when asking nearest-neighbor queries for six different numbers of neighbors over the "*MRHead500*" and "*MRVarious4247*" databases respectively. The numbers of neighbors refer to portions of the database, varying from 0.5% to 15% of each image database. That is, to ask k-NN queries requiring 1% of the database, means to ask NN queries with k=5 for the "*MRHead500*" database, and with k=43 for the "*MRVarious4247*". The curves present the average value when asking 50 queries for each number of neighbors, with the center image randomly chosen from images in the database. In these experiments, each query is submitted to both Slim-trees: the one indexing the normalized histogram, and the other indexing the metric

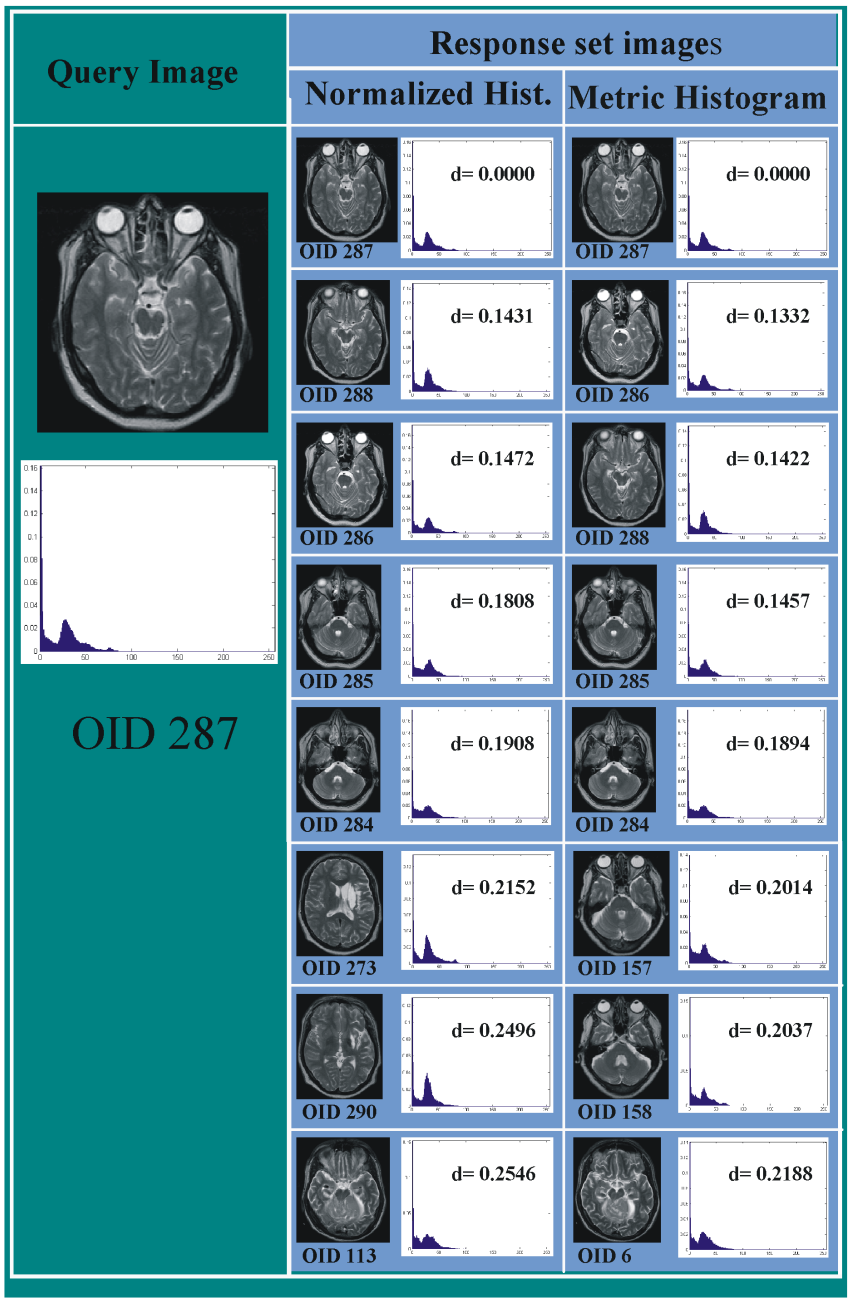

Figure 6. Results of a 8-NN query over the image 287 of the "*MRHead500*". The center column shows the result generated through the normalized histograms. The right column shows the results generated through the metric histograms. The object identifier (OID) of each image and the distance to the query image are also provided.

histogram. Each answer set is sorted by the similarity of each histogram to that of the center of the query. The plots are calculated verifying that increasing numbers of histograms in the result set of the tree indexed by the metric histogram is in the answer set of the tree indexed by the normalized histogram. As it can be seen, the retrieval precision is high, always over 60% even for the "*MRVarious4247*" heterogeneous dataset. With respect to the more homogeneous "*MRHead500*" dataset, the results are always better than 70% of precision, even for queries retrieving 0.5% of the database with 100% of recall.

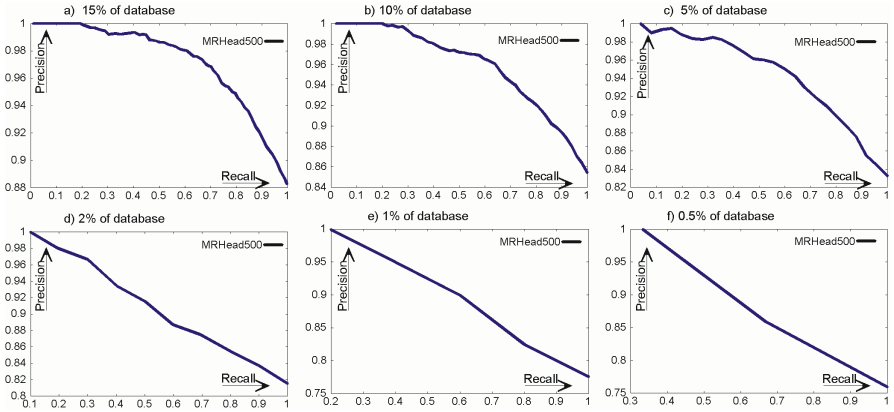

Figure 7. "*MRHead500*" database: Precision vs. Recall plots when answering kNN-queries where the number k is taken as a percentage of the database size. (a) 15%, (b) 10%, (c) 5%, (d) 2%, (e) 1%, (f) 0.5%.

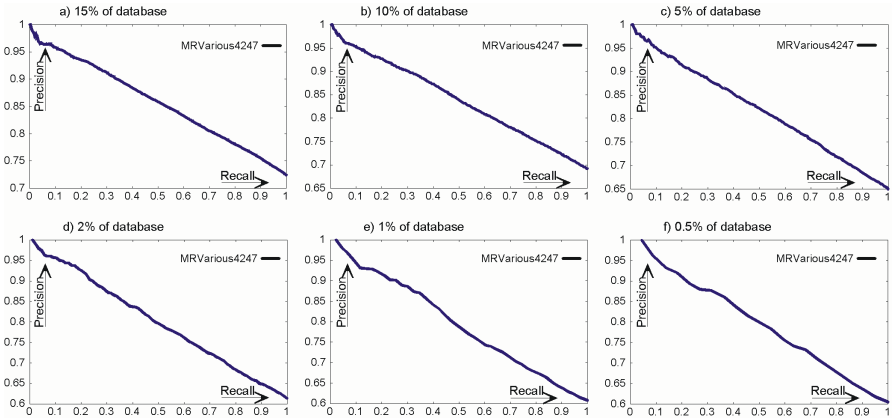

Figure 8. "*MRVarious4247*" database: Precision vs. Recall plots when answering kNN-queries where the number k is taken as a percentage of the database size. (a) 15%, (b) 10%, (c) 5%, (d) 2%, (e) 1%, (f) 0.5%.

5.1.2. *How Scalable and Fast the Metric Histograms are for Indexing*
In this section two questions regarding the scalability of the new proposed histograms are tackled:

(a) How much is the time difference for indexing the image databases through conventional and metric histograms?

(b) Is the difference in time relevant when answering queries over the normalized and the metric histograms?

Table 5.1.2 reports the wall-clock times for building the Slim-tree in order to index both the normalized (conventional) and the metric histograms. It can be seen that it is much faster to build the index over metric histograms (58% faster for the small dataset, and 690% faster for the larger dataset). The additional time needed to create the metric histograms has the same order of magnitude of the time to build the index tree using the metric histogram. However, the tree creation is done only once and the time spent to do it is small when compared to the time to answer queries.

Table II. Wall-clock time to build the slim-tree using metric or normalized histograms from the MRHead500 and the MRVarious4247 image datasets.

Database	Histogram	Time (in seconds)
MRHead500	Metric	462
	Conventional	731
MRVarious4247	Metric	3376
	Conventional	26679

We also measured the number of distance calculations performed per second, and found 1,289,400 MHD distances per second and 269,430 Manhattan distances per second. That is, the calculation of one MHD distance is in average 4.7 times faster than one Manhattan distance over a pair of 256-element arrays.

The largest gain of time when using metric histograms is to answer queries. Figures 9(a) and 9(b) show the total times needed to answer 50 *k-NN* queries, when the value of k is specified corresponding to percentages of the database. Thus, the numbers are proportional to the database size and the results can be compared to different database sizes. Figure 9(a) shows the times to answer queries on the "*MRHead500*" database, and Figure 9(b) shows the times to answer queries on the "*MRVarious4247*" database. As it can be seen, the gain ranges from 4 times faster for small k (0,5% of the database) to more

than 10 times faster for larger portions of the database (15% of the database). All measurements were taken using an Intel Pentium II 450 MHz computer running Windows NT. The software was implemented in Borland C++ language.

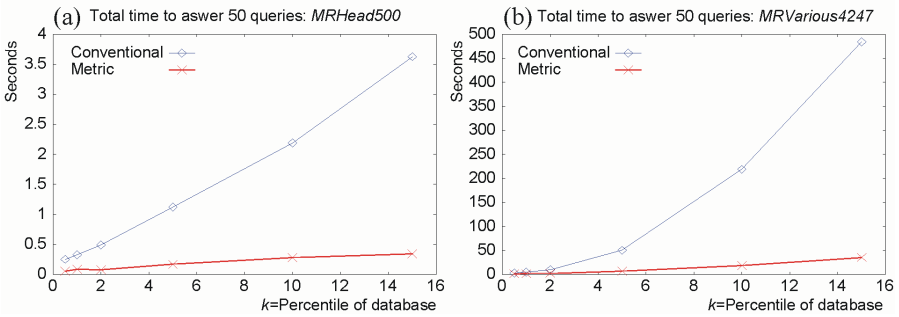

Figure 9. Total time to answer 50 nearest-neighbors queries for each portion of the dataset: 0.5, 1, 2, 5, 10 and 15%, using the conventional and the metric histograms. (a)"*MRHead500*" dataset; (b) "*MRVarious4247*"dataset

5.1.3. *Comparing Images with Different Spatial Resolution*
The second set of experiments was aimed at essaying the effectiveness of using the metric histogram as a way to compare images regardless of its spatial resolution. The objective of these experiments was to test whether the metric histogram is a good choice to execute similarity queries over the WWW, where a first filtering operation could be made using images represented by their thumbnails. Using thumbnails and panels with many thumbnails occurs frequently as front end pages to access the real images in the web or digital libraries.

In this set of experiments we generated three sets of 500 thumbnails from the "*MRHead500*" dataset, and named them "*ThumbBMPNN*", "*ThumbBMPBICUB*", and "*ThumbJPEG*". Each thumbnail dataset was generated preserving the aspect ratio of the original image and reducing its size to 25%, but maintaining the 256 brightness levels. The dataset "*ThumbBMPNN*" was stored in the BMP file format, and generated using the simplest reduction technique, the nearest-neighbor reduction method, to decrease the size of the original images. The dataset "*ThumbBMPBICUB*" was also stored in the BMP file format, but generated using the Bicubic reduction method, which generated new values by interpolation, to decrease the original images. Figure 10 illustrates the exhibit of two thumbnails extracted from the same image, each of which using a different generation method. The "*ThumbJPEG*" dataset was stored in the JPEG file format, and generated using the nearest-neighbor reduction method to generate the thumbnails.

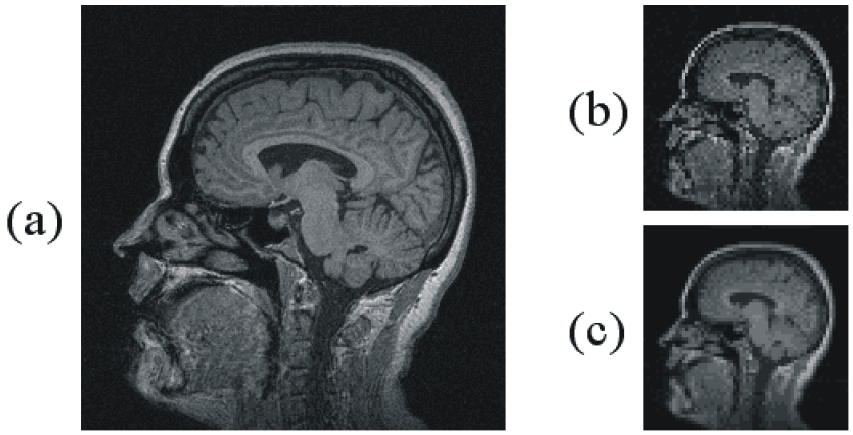

Figure 10. (a) Original image in full size. (b) Thumbnail generated through nearest-neighbor reduction. (c) Thumbnail generated through bicubic reduction.

As in the first set of experiments, we extracted the normalized histogram and generated the metric histogram of each image, creating two sets of features for each dataset of thumbnails. The next step was to index the six feature sets (three datasets and two features - normalized and metric histogram for each dataset), generating six new Slim-trees. Then, we searched the original "*MRHead500*" dataset and compared the results to each thumbnail datasets. Each dataset has two Slim-trees associated with, each one corresponding to a feature, therefore each set of tests involved four datasets, as is illustrated in figure 11. Each ellipse in this figure represents a feature set from an image dataset: "Orig Normal" and "Orig Metric" represent the normalized and metric histograms from the images in the "*MRHead500*" dataset, and "Thumbs Normal" and "Thumbs Metric" represent the normalized and metric histograms from the images in the thumbnail dataset ("*ThumbBMPNN*", "*ThumbBMPBICUB*" or "*ThumbJPEG*") being considered. Five searching tests were performed among the set of four datasets, each of which being represented by one tagged arrow (A to E) in figure 11. For each searching we have considered one feature dataset as correct, and plotted the precision vs. recall graphs considering the other dataset as the target. In figure 11, each arrow starts in the feature dataset considered correct, and ends in the target one. The four searching operations performed are described as follows.

A - Normalized histograms from the "*MRHead500*" dataset compared to metric histograms of the same dataset - this is the same retrieval test performed in the first set of experiments, and is plotted here as a reference for the other curves.

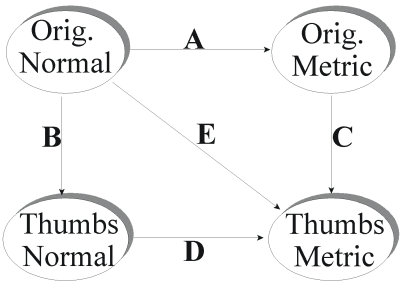

Figure 11. Matching diagram for figures 12, 13 and 14. Each arrow starts at the feature dataset considered correct and ends at the one being checked in each match.

B - Normalized histograms from the "*RHead500*" dataset compared to normalized histograms of one thumbnails dataset - this test shows the usefulness of the image retrieval operations given by the normalized histograms of the thumbnails in place of the normalized histograms of the real images.

C - Metric histograms from the "*MRHead500*" dataset compared to metric histograms of one thumbnails dataset - this test shows how the usefulness of the image retrieval operations given by the metric histograms of the thumbnails in place of the metric histograms of the real images.

D - Normalized histograms from the thumbnails dataset compared to metric histograms of the same dataset - this test shows the usefulness of the image retrieval operations given by the metric histograms of the thumbnails in place of its normalized histograms.

E - Normalized histograms from the "*MRHead500*" dataset compared to metric histograms of one thumbnails dataset - this test shows the usefulness of the image retrieval operations given by the metric histograms of the thumbnails in place of the normalized histograms of the real images.

We show the precision vs. recall graphs obtained for these five retrievals for each of the "*ThumbBMPNN*", "*ThumbBMPBICUB*" and "*ThumbJPEG*" thumbnails dataset respectively in figures 12, 13 and 14. The curves shown are the average obtained for 50 nearest-neighbor queries asking for 5% of the database (25 images each query).

Analyzing the results obtained for the thumbnails stored in BMP format files, we can see from figures 12 and 13 that the thumbnails extracted using the bicubic reduction represents a better choice than those extracted using the nearest-neighbor reduction as a replacement for the original image when performing image searching in databases.

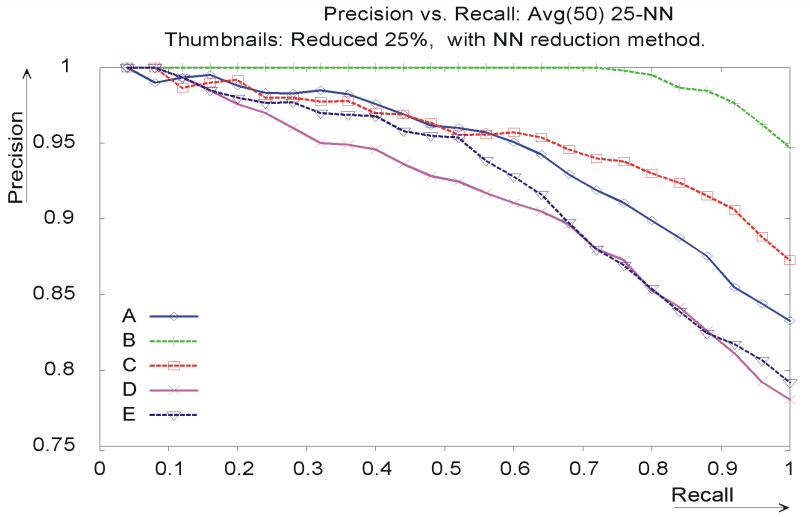

Figure 12. Comparing the "*MrHead500*" and its thumbnails (reduced 25% with nearest-neighbors reduction method) in BMP format.

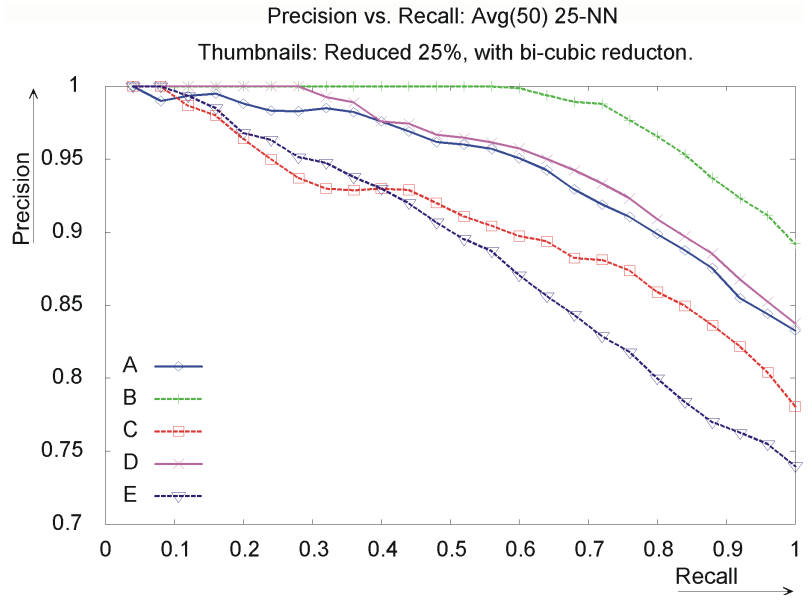

Figure 13. Comparing the "*MrHead500*" and its thumbnails (reduced 25% with bicubic reduction method) in BMP format.

This result corresponds to what one could expect when comparing the exhibit of thumbnails extracted using the nearest-neighbor and the bi-cubic reduction methods as shown in figure 10. Curves A and D of figure 13 shows that comparing full images on the corresponding thumbnails

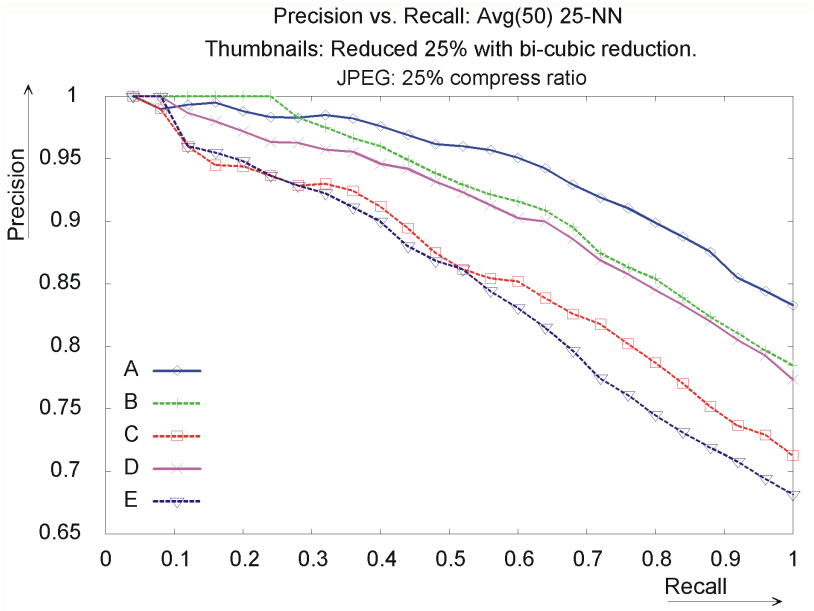

Figure 14. Comparing the "*MrHead500*" and its thumbnails (reduced 25% with bicubic reduction method) in JPEG format with 25% compression rate.

through its metric histograms have almost the same results. Curves A and D of figure 12 shows that using the full size image provides (slightly) better results. However, curves D and E of figure 12 shows that comparing full size images through metric histograms gives almost the same retrieval precision as comparing its thumbnails with metric histograms, showing that the poorer discrimination power of curve D is due to the poorer quality of the NN reduction technique, as the corresponding D and E curves of figure 13 shows that comparing full size images through metric histograms is better than comparing its thumbnails with metric histograms. Nonetheless, both NN and bicubic methods to create thumbnails present sufficient discrimination power for image retrieval, so the thumbnails can be used in place of the real images.

Comparing the results obtained for the thumbnails stored in BMP and in JPEG file format, as shown in figures 13 and 14, we can see that both file formats present similar results. The thumbnails for both file formats were created using the bicubic reduction method, but in the JPEG file format they were stored using a 25% compression rate, so it is a lossy representation of images. This can be seen in figure 14, where curve B (comparing the original images to the full size images in JPEG format) shows that the lossy storage of the images makes it better to compare the original images through the metric histogram (curve A).

In fact, curve D and A shows that comparing the thumbnails in BMP format with the metric histogram is the same as comparing the full size images in JPEG format. This is a very good result, because the majority of images in the web and in many digital libraries are stored in JPEG or other lossy file format, to reduce transmission times.

6. Related Work

A taxonomy of the features most used in the analysis of images was presented in [11]. The author classified them into five groups: raw intensity, border detection, salient features, statistical features and high level features. Histograms are classified as statistical features, and they have been long ago used as an initial step to filter out interesting images from large collections in automated tools [12] [27] [29] [49]. In fact, almost every image retrieval system relies on histogram-based filtering to trim down the number of images that are submitted to the user, or to more elaborated analysis processes. This happens not only in general purpose systems such as QBIC [21], Chabot [38] and VisualSEEk [50], but also in specific domains, such as ASSERT [46] for radiology and WebMars [39] for web browsers, among others. All systems rely on some form of indexing structure to process the queries.

Aiming at improving indexing performance, researchers have tried to use the fewer possible features, yet maintaining good query discrimination. Regarding the use of histograms, this objective is being pursued with a careful selection of the features, as well as using dimensionality reduction techniques. An interesting example of the former technique is the use of HSV (hue, saturation and value) instead of RGB (red, green and blue) colors, what some researchers considers more similar to the human interpretation of colored images [3]. However, the majority of researches have been done in finding more concise representations of histograms using dimensionality reduction techniques [12]. Standard dimensionality reduction techniques like SVD has been employed [31] with good results, as well as neural networks [35]. Image domain transformations such as FFT and wavelets have also been reported [20] [37] [2]. Kuo and Chang proposed a histogram approximation through the use of multiple Gaussian probability density function [34], where each histogram is approximated by a predefined number k of Gaussians, and a pair of histograms is compared averaging the difference of its mean, standard deviation and skewness. All these attempts leaded to histogram representations with reduced number of features, but always with a predefined number of dimensions not benefitting from the cor-

relations among close bins at each image. So the particularities of each individual image cannot be taken in account.

In fact, almost all distance functions used to compare images through they feature spaces use some form of vector comparisons, for example, L_p-norms, χ-Square, Kolmogorov-Smirnov, etc. The main contribution of this paper is to consider histograms as curves, comparing a pair of histograms through the difference in the area covered by both curves. This approach frees the representation of each image from the representations of the other images in the dataset, allowing using the best representation of each of which, yet allowing them to be compared.

7. Conclusions

Images are one of the most popular data present in the web. This is mainly due to the powerful representation of semantic information given by images. Allied to the fact that the web has been used as a universal data repository, it is important to search and retrieve information contained in images to attend the questions placed by the users. However, we need to remember that comparing two images by content is a very time consuming process. As the image database grows, the cost of comparing images grows at the same pace, or even at a superlinear pace, making this a burdensome and costly process. Therefore, to reduce the cost of searching in a dataset, the CBIR systems should take advantage of indexed access methods. If the users intention is to ask similarity queries, the metric access methods are the more suitable ones.

When queries are issued over an image database to retrieve information based on the image content, a filtering process takes place aiming to reduce the number of images to be compared. This filtering operation uses features extracted from the images through relatively inexpensive algorithms. One of the features most frequently used in the early filtering stages is based on the image histograms.

The main objective of this paper is to define a faster way to execute the filtering process over an image database based on the image histograms. To achieve this target we proposed a new technique, called the "metric histogram". We showed that by using this technique the filtering step can be improved, achieving up to 10 times faster image selection for similarity queries. Moreover, the creation of index structures using metric histograms is at least 58% and up to 700% times faster as compared to structures created using the conventional histograms.

Metric histograms are defined in a metric domain, so the parameters that best describes one histogram can be used without compromising the whole set of images. To allow using metric access methods to answer similarity queries, we defined the $MHD()$ distance function, which complies with the properties of a metric distance function. This function is also a faster way to compare histograms, as it can be executed in average 4.7 times faster than using the Manhattan distance function over 256-element vectors.

Metric histograms also have some very desirable side effects: it allows the retrieval of images in a way that is invariant to scale, rotation and translation of the objects in the image, and is also invariant to brightness changes of the image. These effects enable that images similar to the target one, but with different brightness levels, scale, and placement can also be retrieved without further computational effort. Moreover, we showed that the proposed technique also works well when comparing full-sized images with spatial-reduced resolution images (thumbnails), even when the images are compressed in lossy JPEG format. That is, the discrimination power of the metric histogram remains high when comparing the images by its thumbnails. This last property is especially interesting in web searching applications, as it is very common that sites which export many images usually use thumbnail pages as a front end interface to provide users a initial way to select images, reducing transmission costs.

Acknowledgements

This research was supported, in part, by the scholarship grants from the Sao Paulo State Research Foundation (FAPESP) numbers 01/12536-5, 01/14.578-7 and 02/07318-1, and by the Brazilian National Research Council (CNPq) under grants 52.1685/98-6, 860.068/00-7 and 35.0852/94-4.

References

1. Albanesi, M., S. Bandelli, and M. Ferretti: 2001, 'Quantitative assessment of qualitative color perception in image database retrieval'. In: *11th Intl. Conference on Image Analysis and Processing*. Thessaloniki, Greece, pp. 410–415.
2. Albuz, E., E. Kocalar, and A. A. Khokhar: 2001, 'Scalable Color Image Indexing and Retrieval Using Vector Wavelets'. *IEEE Transactions on Knowledge and Data Engineering* **13**(5), 851–861.

3. Androutsos, D., K. N. Plataniotis, and A. N. Venetsanopoulos: 1999, 'A Novel Vector-Based Approach to Color Image Retrieval Using a Vector Angular-Based Distance Measure'. *Computer Vision and Image Understanding* **75**(1/2), 46–58.
4. Araujo, M. R. B., J. Traina, Caetano, A. J. M. Traina, J. M. Bueno, and H. L. Razente: 2002, 'Extending Relational Databases to Support Content-based Retrieval of Medical Images'. In: *IEEE International Conference on Computer Based Medical Systems - CBMS*. Maribor, Slovenia, pp. 303–308.
5. Aslandogan, Y. A. and C. T. Yu: 1999, 'Techniques and Systems for Image and Video Retrieval'. *IEEE Transactions on Knowledge and Data Engineering* **11**(1), 56–63.
6. Baeza-Yates, R. and B. A. Ribeiro-Neto: 1999, *Modern Information Retrieval*. Wokingham, UK: Addison-Wesley.
7. Beckmann, N., H.-P. Kriegel, R. Schneider, and B. Seeger: 1990, 'The R*-tree: An Efficient and Robust Access Method for Points and Rectangles'. In: *ACM Int'l Conference on Data Management (SIGMOD)*. pp. 322–331.
8. Berchtold, S., D. A. Keim, and H.-P. Kriegel: 1996, 'The X-tree: An Index Structure for High-dimensional data'. In: *Intl. Conf. on Very Large Databases (VLDB)*. Bombay, India, pp. 28–39.
9. Berman, A. and L. G. Shapiro: 1998, 'Selecting Good Keys for Triangle-Inequality-Based Pruning Algorithms'. In: *Intl. Workshop on Content-Based Access of Image and Video Databases (CAIVD '98)*. Bombay, India, pp. 12–19.
10. Bozkaya, T. and Z. M. zsoyoglu: 1999, 'Indexing Large Metric Spaces for Similarity Search Queries'. *ACM Transactions on Database Systems (TODS)* **24**(3), 361–404.
11. Brown, L. G.: 1992, 'A Survey of Image Registration Techniques'. *ACM Computing Surveys* **24**(4), 325–376.
12. Brunelli, R. and O. Mich: 1999, 'On the Use of Histograms for Image Retrieval'. In: *IEEE Intl. Conf. on Multimedia Computing and Systems (ICMCS)*, Vol. 2. Florence, Italy, pp. 143–147.
13. Bueno, J. M., F. Chino, A. J. M. Traina, J. Traina, Caetano, and P. M. d. A. Marques: 2002, 'How to Add Content-based Image Retrieval Capability in a PACS'. In: *IEEE International Conference on Computer Based Medical Systems - CBMS*. Maribor, Slovenia, pp. 321–326.
14. Burkhard, W. A. and R. M. Keller: 1973, 'Some Approaches to Best-Match File Searching'. *Communications of the ACM* **16**(4), 230–236.
15. Chakrabarti, K. and S. Mehrotra: 1999, 'The Hybrid Tree: An Index Structure for High Dimensional Feature Spaces'. In: *Intl. Conf. on Data Engineering (ICDE)*. Sydney, Australia, pp. 440–447.
16. Chvez, E., G. Navarro, R. A. Baeza-Yates, and J. L. Marroqun: 2001, 'Searching in Metric Spaces'. *ACM Computing Surveys* **33**(3), 273–321.
17. Ciaccia, P., M. Patella, and P. Zezula: 1997, 'M-tree: An efficient access method for similarity search in metric spaces'. In: M. Jarke (ed.): *Intl. Conf. on Very Large Databases (VLDB)*. Athens, Greece, pp. 426–435.
18. Comer, D.: 1979, 'The Ubiquitous B-Tree'. *ACM Computing Surveys 11(2):* **11**(2), 121–137.
19. Esperança, C. and H. Samet: 1997, 'A Differential Code for Shape Representation in Image Database Application'. In: *Intl. Conf. Image Processing*.
20. Faloutsos, C.: 1996, *Searching Multimedia Databases by Content*. Boston, MA: Kluwer Academic Publishers.

21. Flickner, M. and e. alli: 1995, 'Query by Image and Video Content: The QBIC System'. *IEEE Computer* **28**(9), 23–32.
22. Gaede, V. and O. Gnther: 1998, 'Multidimensional Access Methods'. *ACM Computing Surveys* **30**(2), 170–231.
23. Gagaudakis, G. and P. L. Rosin: 2001, 'Shape measures for image retrieval'. In: *Intl. Conference on Image Processing*. Thessaloniki, Greece, pp. 757 –760.
24. Garcia-Molina, H., J. D. Ulhman, and J. Widow: 2000, *Database System Implementation*. Prentice Hall.
25. Gimel'farb, G. L. and A. K. Jain: 1996, 'On Retrieving Textured Images from an Image Database'. *Pattern Recognition* **29**(9), 1,461–1,483.
26. Gntzer, U., W.-T. Balke, and W. Kiessling: 2000, 'Optimizing Multi-Feature Queries for Image Databases'. In: *Intl. Conf. on Very Large Databases (VLDB)*. Cairo - Egypt, pp. 419–428.
27. Gupta, A. and S. Santini: 2000, 'Toward feature Algebras in Visual Databases: The Case for a Histogram Algebra.'. In: H. Arisawa and T. Catarci (eds.): *Fifth Working Conference on Visual Database Systems (VDB5)*, Vol. 168. Fukuoka, Japan.
28. Guttman, A.: 1984, 'R-Tree : A dynamic Index Structure for Spatial Searching'. In: *ACM Int'l Conference on Data Management (SIGMOD)*. Boston, MA, pp. 47–57.
29. Hafner, J., H. Sawhney, W. Equitz, M. Flickner, and W. Niblack: 1995, 'Efficiente Color Histogram Indexing for Quadratic Form Distance Function'. *IEEE Transactions on Patterns Analysis and Machine Intelligence* **17**(7), 729–736.
30. Kamel, I. and C. Faloutsos: 1994, 'Hilbert R-tree: An Improved R-tree using Fractals'. In: J. B. Bocca, M. Jarke, and C. Zaniolo (eds.): *20th Intl.Conf. on Very Large Data Bases (VLDB)*. Santiago del Chile, Chile, pp. 500–509.
31. Kanth, K. V. R., D. Agrawal, A. El Abbadi, and A. K. Singh: 1999, 'Dimensionality Reduction for Similarity Searching in Dynamic Databases'. *Computer Vision and Image Understanding* **75**(1/2), 59–72.
32. Ko, B., H.-S. Lee, and H. Byun: 2000, 'Image Retrieval Using Flexible Image Subblocks'. In: *ACM Symposium on Applied Computing*, Vol. 2. pp. 574 – 578.
33. Korn, F., B.-U. Pagel, and C. Faloutsos: 2001, 'On the 'Dimensionality Curse' and the 'Self-Similarity Blessing"'. *IEEE Transactions on Knowledge and Data Engineering* **13**(1), 96–111.
34. Kuo, W.-J. and R.-F. Chang: 2000, 'Approximating the statistical distribution of color histogram for content-based image retrieval'. In: *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '00*, Vol. 4. pp. 2007 –2010.
35. Lim, J.-H., J. K. Wu, S. Singh, and D. Narasimhalu: 2001, 'Learning Similarity Matching in Multimedia Content-Based Retrieval'. *IEEE Transactions on Knowledge and Data Engineering* **13**(5), 846–850.
36. Lin, K.-I. D., H. V. Jagadish, and C. Faloutsos: 1994, 'The TV-Tree: An Index Structure for High-Dimensional Data'. *VLDB Journal* **3**(4), 517–542.
37. Mandal, M. K., S. Panchanathan, and T. Aboulnasr: 1998, 'Fast Wavelet Histogram Techniques for Image Indexing'. In: *IEEE Workshop on Content-based Access of Image and Video Libraries*. Santa Barbara, CA.
38. Ogle, V. E. and M. Stonebraker: 1995, 'Chabot: Retrieval from a Relational Databases of Images'. *IEEE Computer* **28**(9), 40–48.
39. Ortega-Binderberger, M., S. Mehrotra, K. Chakrabarti, and K. Porkaew: 2000, 'WebMARS: A Multimedia Search Engine for Full Document Retrieval and

Cross Media Browsing'. In: *Multimedia Information Systems Workshop 2000 (MIS2000)*. Chicago, IL.
40. Pass, G., R. Zabih, and J. Miller: 1996, 'Comparing Images Using Color Coherence Vector'. In: *ACM Multimedia*. Boston, MA, pp. 65–73.
41. Pentland, A., R. Picard, and S. Sclaroff: 1996, 'Content-based Manipulation of Image Databases'. *Intl. Journal of Computer Vision* **18**(3), 233–254.
42. Robinson, J. T.: 1981, 'The K-D-B-Tree: A Search Structure For Large Multidimensional Dynamic Indexes'. In: Y. E. Lien (ed.): *ACM International Conference on Management of Data - SIGMOD*. SIGMOD Conference 1981:, pp. 10–18.
43. Samet, H.: 1995, 'Spatial data structures in Modern Database Systems: The Object Model, Interoperability, and Beyond'. Addison-Wesley/ACM Press, pp. 361–385.
44. Santos, Roberto Figueira, F., A. J. M. Traina, J. Traina, Caetano, and C. Faloutsos: 2001, 'Similarity Search without Tears: The OMNI Family of All-purpose Access Methods'. In: *Intl. Conf. on Data Engineering (ICDE)*. Heidelberg, Germany, pp. 623–630.
45. Sellis, T. K., N. Roussopoulos, and C. Faloutsos: 1987, 'The R+-tree: A Dynamic Index for Multi-dimensional Objects'. In: *Intl. Conf. on Very Large Databases (VLDB)*. Brighton, England, pp. 507–518.
46. Shyu, C.-R., C. E. Brodley, A. C. Kak, A. Kosaka, A. M. Aisen, and L. S. Broderick: 1999, 'ASSERT: A Physician-in-the-Loop Content-Based Retrieval System for HRCT Image Databases'. *Computer Vision and Image Understanding* **75**(1/2), 111132.
47. Siegel, E. L.: 1999, 'Current State of the Art and Future Trends'. In: E. L. Siegel and R. M. Kolodner (eds.): *Filmless Radiology*. New York City, NY: Springer Verlag, pp. 3–20.
48. Siegel, E. L. and R. M. Kolodner: 1999, *Filmless Radiology*. New York City, NY: Springer Verlag.
49. Smeulders, A. W. M., M. Worring, S. Santini, A. Gupta, and R. Jain: 2000, 'Content-Based Image Retrieval at the End of the Early Years'. *IEEE Transactions on Patterns Analysis and Machine Intelligence* **22**(12).
50. Smith, J. R. and S.-F. Chang: 1996, 'VisualSEEk: a fully automated content-based image query system'. In: *ACM Multimedia '96*. Boston, MA, pp. 87–98.
51. Tomita, F. and T. Saburo: 1990, *Computer Analysis of Visual Textures*. Kluwer.
52. Traina, A. J. M., J. Traina, Caetano, J. M. Bueno, and P. M. d. A. Marques: 2002a, 'The Metric Histogram: A New and Efficient Approach for Content-based Image Retrieval'. In: *Sixth IFIP Working Conference on Visual Database Systems*. Brisbane, Australia, pp. 297–311.
53. Traina, Caetano, J., A. J. M. Traina, C. Faloutsos, and B. Seeger: 2002b, 'Fast Indexing and Visualization of Metric Datasets Using Slim-trees'. *IEEE Transactions on Knowledge and Data Engineering* **14**(2), 244–260.
54. Traina, Caetano, J., A. J. M. Traina, B. Seeger, and C. Faloutsos: 2000, 'Slim-Trees: High Performance Metric Trees Minimizing Overlap Between Nodes'. In: C. Zaniolo, P. C. Lockemann, M. H. Scholl, and T. Grust (eds.): *Intl. Conf. on Extending Database Technology*, Vol. 1777 of *Lecture Notes in Computer Science*. Konstanz, Germany, pp. 51–65.
55. Tuytelaars, T. and L. J. V. Gool: 1999, 'Content-based Image Retrieval Based on Local Affinely Invariant Regions'. In: D. P. Huijsmans and A. W. M. Smeulders (eds.): *Third International Conference on Visual Information and*

Information Systems - VISUAL'99, Vol. 1614 of *Amsterdam, The Netherlands*. Amsterdam, The Netherlands, pp. 493–500.
56. Vailaya, A., M. A. T. Figueiredo, A. K. Jain, and H.-J. Zhang: 2001, 'Image classification for content-based indexing'. *IEEE Transactions on Image Processing* **10**(1), 117–30.
57. Yamamoto, H., H. Iwasa, N. Yokoya, and H. Takemura: 1999, 'Content-based Similarity Retrieval of Images Based on Spatial Color Distributions'. In: *10th Intl. Conference on Image Analysis and Processing*. pp. 951–956.
58. Zhou, F., J. F. Feng, and Q. Y. Shi: 2001, 'Texture feature based on local Fourier transform'. In: *International Conference on Image Processing*, Vol. 2. Thessaloniki, Greece, pp. 610 –613.