WINDSURF: a Region-Based Image Retrieval System

Ilaria Bartolini, Paolo Ciaccia, Marco Patella
DEIS - CSITE-CNR, University of Bologna, Italy
e-mail: {ibartolini,pciaccia,mpatella}@deis.unibo.it

July 6, 2001

Abstract

Content-based image retrieval systems characterize images by means of relevant *features*, such as color distribution and texture information, and then use such feature values to determine those images which are most similar to a query image. However, this approach is not adequate when images have a complex, not homogeneous, structure, in which case using *global features* leads to inaccurate content descriptions. In this paper we present WINDSURF, an image retrieval system that implements a new approach to content-based image search. WINDSURF applies the wavelet transform to extract color and texture features from an image, and then partitions the image into a set of "homogeneous" regions, each described by a set of *local features*. Similarity between images is then assessed by first computing similarity scores between regions and then combining the results at the image level. Remarkably, WINDSURF image similarity model only requires that the overall similarity score between two images is a monotonic function of the similarity scores of matching regions. From the query processing point of view, we introduce a novel index-based algorithm that can make use of any distance-based access method to retrieve the k best-matching images for a given query. It turns out that this is the *first* correct algorithm for region-based image similarity queries. Experimental results on a medium-size image data-set demonstrate the effectiveness and the efficiency of our approach.

1 Introduction

The advent of multimedia age poses a number of new challenges to database researchers. In particular, digital image libraries require effective and efficient automated retrieval based on the "semantic" content of images. The boost of graphics capabilities in modern computer systems and the growing of Internet have further contributed to the increased availability of digital images. A classical method to characterize the content of images is to have human experts manually annotating each image with a textual description, so that text-based information retrieval techniques can be applied [Sal88]. However, this approach is clearly impracticable in case of very large image databases and its effectiveness is highly dependent on the subjective opinions of the experts, who are also likely to supply different descriptions for a same image [OS95].

Consequently, the approach taken by modern content-based image retrieval (CBIR) systems is to define a set of relevant properties (*features*) able to effectively characterize the content

of images and then to use these features for retrieval purposes [GR95]. Such features should be "simple enough" to allow the design of automatic extraction algorithms, yet "meaningful enough" to capture the image content. To this end, recent studies have highlighted the fact that *global features*, like color and texture, indeed possess a rich semantic value, and as such they are used by several CBIR systems [FEF$^+$94, SO95, PPS96, SC96]. Under this view each image is typically represented by a high-dimensional *feature vector*, whose dimensionality depends on the number and on the type of extracted features, and similarity between images is assessed by defining a suitable distance function on the resulting feature space [Fal96].

It is a fact that CBIR systems that rely on global features cannot support queries like, say, "Find all the images containing a small red region under a big blue region" that refer to *local* properties of the images. Thus, the need to extract not only global but also *local features* has emerged, and a number of *region-based* image retrieval systems, that fragment each image into a set of homogeneous regions, have been presented [SC96, CTB$^+$99, NRS99]. In region-based systems, similarity assessment between images is performed by associating regions in the query image with those contained in database images and by taking into account similarity between associated regions. To this end, features are extracted for each region and a distance function is used to compare regions' descriptors. Existing systems, however, either consider a scenario, which is beyond the scope of the present work, where also spatial constraints are taken into account [BDV99], or use naïve heuristic matching algorithms which are not guaranteed to return the correct results. As an example, suppose that a user looks for images containing two tigers. In this case the query image will contain (at least) two regions, each representing a tiger. If a database (DB) image contains a single "tiger" region, clearly it is not correct to associate both query regions to the single tiger region of the DB image. However, as we will argument in Section 2, this can easily happen with current region-based systems.

In the following we will focus our attention on the processing of k nearest neighbors (best-matches) queries, where the user asks for the k images in the DB which are most similar, according to the similarity measure implemented by the CBIR system, to the query image. Range queries, where the user has to specify a minimum similarity threshold α that images have to exceed in order to be part of the result, are not well suited for the scenario we envision. In fact, since the user has no a priori knowledge on the distribution of similarities between images, he/she has no way to guess the "right" value for α. Indeed, a high α value can easily lead to an empty result, whereas slightly decreasing α could result in an overwhelming number of returned images. This situation is further complicated in region-based retrieval, where more than one threshold could be required (see Section 2.1).

In this paper we present the WINDSURF system (Wavelet-based INDexing of imageS Using Region Fragmentation), a new region-based image retrieval system. WINDSURF applies the wavelet transform to extract color and texture features from an image, and then clusters wavelet coefficients so as to obtain a set of "homogeneous" regions, each characterized by its own local features. After providing a *correct* definition of *image similarity* under the region-based retrieval model, we introduce a novel index-based algorithm, called $\mathcal{A}_0^{WS}$, suitable for any distance-based access method, to obtain the k best-matching images for a given query. It turns out that this is the *first* correct algorithm for region-based image similarity queries. Experimental results on

a medium-size image data-set demonstrate the effectiveness and the efficiency of our approach, as also compared to alternative retrieval strategies.

The rest of the paper is organized as follows. In Section 2 we describe existing CBIR systems, and outline their limits from the query processing point of view. Section 3 introduces the WINDSURF system. In Section 4 we formalize the problem of computing similarity between images as a generalized assignment problem and present a simple sequential algorithm (`ERASE`) for its resolution. In Section 5 we describe the $\mathcal{A}_0^{WS}$ index-based algorithm. Experimental analysis of the proposed solutions is in Section 6, where we consider both efficiency and effectiveness issues. Section 7 concludes and suggests possible directions for future work.

2 Background

Many CBIR systems have been designed and developed over the last years. What can be called the first generation of CBIR systems used *global features* to characterize the images content. For example, QBIC [FEF+94], developed at the IBM Almaden Research Center, extracts from each image a number of features, namely color, texture, and shape. Color is represented by means of histograms that are compared using a distance function that also takes into account the similarity between different colors (*cross-talk*). Texture is analyzed globally by extracting information on *coarseness*, *contrast*, and *direction*. Similarity between images is computed using a weighted Euclidean distance on the overall extracted vector.

Stricker and Orengo [SO95] propose a different approach to color similarity, where the first three moments of the distribution of each color channel are considered. Thus, each image is represented by a 9-dimensional feature vector, and a simple weighted Manhattan distance is used to compare images.

The Photobook system developed by MIT Media Lab [PPS96] uses a stochastic model (*Wold-decomposition*) to assess the similarity between images based on texture.

Techniques operating in the time-frequency domain, such as the wavelet transform [Dau92] (see also Appendix A), have also been proposed to obtain a multi-resolution image representation. As an example, the WBIIS system [WWFW97] uses Daubechies' wavelets [Dau92] to derive a 768-dimensional vector of wavelet coefficients that preserve spatial image information. Although this approach offers a better frequency location with respect to other algorithms, it leads to poor results for queries where spatial location and scale of objects is not preserved [NRS99].

All above described approaches (as well as many others not covered here) use global features to represent image semantics, thus they are not adequate to support queries looking for images with specific "objects" with particular color and/or texture (and possibly spatially arranged in a particular way), "partial-match" queries (where only a part of the query image is specified), and shift/scale-invariant queries, where the position and/or the dimension of the seeked objects is not deemed relevant. *Region-based* image retrieval systems aim to overcome these limitations by fragmenting an image into a set of "homogeneous" regions, which can then be described by means of *local features* [SC96, CTB+99, NRS99]. Note that the concept of "homogeneity" is by no means easy to define. For instance, if one considers each pixel separately, texture information

is lost and only "color homogeneity" can be assessed. For a more complex example, consider an image where a flag with red and blue vertical stripes appears. A human would certainly recognize this as a homogeneous region, even if it contains pixels with rather different colors, since he/she sees a repeated pattern.

VisualSEEk [SC96] is an example of region-based system that considers information from both the spatial and the frequency domains in order to decompose each image into regions. The similarity between two images is computed by taking into account color, location, size, and relative positions of regions. Query processing, however, is carried out by using a simple heuristic algorithm. First, for each region of the query image, a range query on color, location, and size is issued with similarity thresholds provided by the user; then, a *candidate set* of images is built, by taking into account only those images that have one region in all the result regions sets; finally, the optimum match is computed on the set of candidate images. It is clear that the use of similarity thresholds has no direct counterparts in a user's mind, and cannot gaurantee that the images most similar to the the query image are retrieved.

2.1 WALRUS

WALRUS (WAveLet-based Retrieval of User-specified Scenes) [NRS99] is a region-based image retrieval system where the similarity measure between a pair of images is defined to be the fraction of the area of the two images covered by matching regions of the two images.

WALRUS pre-processes an image in two steps. First, it generates a set of *sliding windows* with different sizes and computes a "signature" (local feature vector) for each sliding window, where a signature consists of all the coefficients from the lowest frequency band of the Haar wavelet transform applied to the pixels in the window. The next step is to cluster the sliding windows by computing the similarity between their signatures. Each cluster, thus, consists of a set of windows with similar characteristics (i.e. color and texture), which together define a region. Wavelet signatures of the windows in a cluster are then averaged to obtain the region feature vector. To speed-up the retrieval, WALRUS indexes regions' descriptors using an R*-tree [BKSS90].

In order to submit a query to WALRUS, a user has to specify a query image and two similarity thresholds, ϵ and ξ. After extracting regions from the query image, WALRUS uses the index to find all regions in the DB that are similar enough to a query region, that is, regions whose signatures are within ϵ distance from the signature of the query region. Then, similarity between images is assessed by adding up the sizes of matched regions, as obtained from the index serach step, and the result of the query consists of the images for which the similarity with the query image is not lower than the ξ threshold.

From the query processing point of view, the main limitation of WALRUS is that it requires the specification of two similarity thresholds. The choice of the two parameters is not very meaningful, since the user has no clear way to determine what a difference between threshold values actually represents. As already argued in Section 1, we believe that range queries are not suitable for effecive image retrieval.

2.2 Blobworld

Blobworld [CTB+99] is a system that determines coherent image regions that roughly correspond to objects. Blobworld models an image as a set of regions (*blobs*) which are homogeneous with respect to color and texture. Each blob is described by its color distribution and by its mean texture descriptors, obtaining a 220-D feature vector (218-bins color histogram and 2 texture descriptors). Querying is then based on the features of some (typically, one or two) regions of interest, rather than on a description of the whole image.

In the image pre-processing phase, Blobword first extracts pixel features, then it groups similar pixels into blob regions, and finally determines the feature vectors of the blobs. In detail, the pixels distribution is modeled in a 8-D space (L*a*b* descriptors for color, *anisotropy*, *orientation*, and *contrast* for texture, and spatial position of the pixel) using a mixture of two to five Gaussians. To fit the mixture of Gaussians model to the pixel data, the Expectation-Maximization (EM) algorithm is used, whereas the number of Gaussians that best suits the real number of groups contained in the image is determined by means of the Minimum Descriptor Length principle [BCGM98]. Once a model is selected, the system performs a spatial grouping of connected pixels belonging to the same cluster.

At query time, the user composes a query by submitting to the system an image of interest and selecting some of the blobs in the image (an "atomic query" is composed by a single blob, whereas a "compound query" is specified by two or more blobs). When dealing with a compound query, which is the most common case, each blob in the query image is associated to its "best" blob in the DB image under consideration (a quadratic, L_2-like, distance function between the feature vectors is used to this purpose). Then, the overall score is computed by using (weighted) fuzzy-logic operators (conjunctions, disjunction, and negation) applied to the scores of matched blobs. Finally, images are ranked according to their overall score and the k best matches are returned.

In order to speed-up query processing, Blobworld can also use an R-tree-like structure to index the color descriptors of the blob feature vectors [TCH00] (no texture information is taken into account when an index is used). For each blob in the query image, a predetermined number (in the order of the hundreds) of "best matches" is retrieved by using the index. Note that the use of an index can lead to miss the correct best images, since there is no guarantee that such images will be included within those returned by the index itself, as it will be shown in Section 5. Among all the images containing regions obtained in the index-retrieval step, the "true", above described, matching algorithm is then used to obtain the result images. However, since best matches for query blobs are computed by ignoring matches for other blobs, it could be the case that a single blob in a DB image is associated to two distinct query blobs (remind the "two tigers" example in Section 1).

3 The Windsurf system

WINDSURF (*W*avelet-Based *Ind*exing of Image*s* *U*sing *R*egion *F*ragmentation) is a region-based image retrieval system that uses the Discrete Wavelet Transform (DWT) [Dau92] and a *k-means* clustering algorithm to segment an image into regions. Each region is represented by means of

a set of features and the similarity between regions is measured using a specific metric function on such features. Image pre-processing consists of the three steps shown in Figure 1, namely:

DWT The image is analyzed in the time-frequency domain using a 2-D DWT.

Clustering The image is fragmented into a set of regions using a k-means clustering algorithm that groups together similar wavelet coefficients (*clustering features*).

Feature Indexing Regions obtained from the clustering phase are represented by means of a set of *similarity features*.

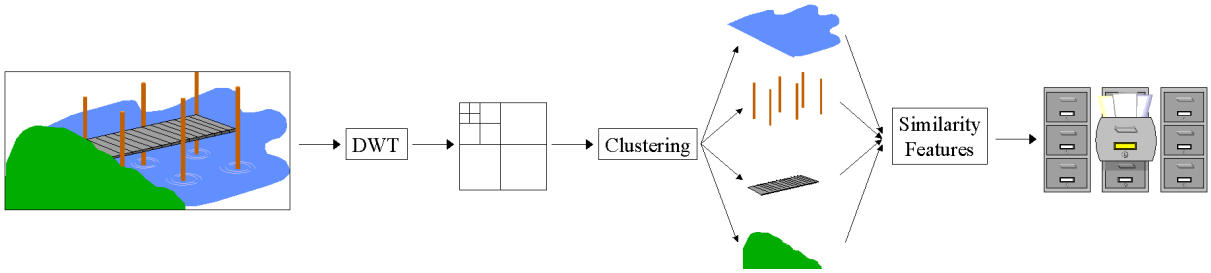


Figure 1: Steps of the WINDSURF image indexing process.

3.1 DWT

WINDSURF views an image as a 2-D signal to be analyzed by means of a 2-D DWT in the time-frequency domain. More in detail, we use Haar wavelets from the WAILI software library [UVJ+97] and represent images in the HSV color space, because in this space each color component is perceptually independent and uniform [Smi97].

The j-th wavelet coefficient of sub-band B ($B \in \mathcal{B} = \{LL, LH, HL, HH\}$, where L stands for "low" and H for "high") and DWT level l is a 3-D vector, i.e.:

$$w_j^{l;B} = (w_{0_j}^{l;B}, w_{1_j}^{l;B}, w_{2_j}^{l;B}) \tag{1}$$

where each component refers to a color channel c ($c \in \{0, 1, 2\}$). The *energy* of $w_j^{l;B}$ on the c and d channels is then defined as:

$$e_{cd_j}^{l;B} = w_{c_j}^{l;B} \cdot w_{d_j}^{l;B} \tag{2}$$

When $c = d$, $e_{cc_j}^{l;B}$ is called the *channel energy* of channel c, whereas when $c \neq d$, $e_{cd_j}^{l;B}$ is termed the *cross-correlation energy* between channels c and d. The energy vector

$$e_j^{l;B} = \left(e_{00_j}^{l;B}, e_{01_j}^{l;B}, e_{02_j}^{l;B}, e_{11_j}^{l;B}, e_{12_j}^{l;B}, e_{22_j}^{l;B}\right) \tag{3}$$

captures both color and texture information through channel and cross-correlation energies, respectively. This is similar to the approach described in [Smi97] and is known to be one of the more robust methods for the representation of texture features [CK93, VSLV].

3.2 Clustering

The aim of the clustering phase is to fragment an image into a set of regions by grouping together image pixels that are similar in color and texture features. To this end, we apply a clustering algorithm to the wavelet coefficients (*clustering features*) obtained by the DWT step. In particular, we apply a *k-means* algorithm with a "validity function" which is a variant of the one proposed for the *fuzzy* k-means algorithm [XB91] (see below).

Given a set $X = \{x_1, \ldots, x_N\}$ of N points (wavelet coefficients, in our case) to be clustered, the k-means algorithm starts with a set of k randomly-chosen *centroids*, $\{\mu_1, \ldots, \mu_k\}$, and then assigns each point x_j to its closest centroid μ_i. After this, the algorithm iterates by recomputing centroids and reassigning the points, until either a stable state or a limit to the number of iterations are reached. It can be proved that k-means algorithm leads to minimize the function

$$J = \sum_{i=1}^{k} \sum_{x_j \in C_i} \delta(x_j, \mu_i)^2 \tag{4}$$

where $\delta(x_j, \mu_i)$ is the distance between x_j and its closest centroid μ_i. Obviously, both the final value of J and the result of the algorithm depend on the value of k and on the choice of the distance function $\delta()$. As for $\delta()$ we use the *Mahalanobis distance* applied to the 3-D wavelet coefficients of the LL sub-band of the 3-rd DWT level, that is, $x_j \equiv w_j^{3;LL}$. This choice is due to the results of extensive experimental evaluation, which demonstrated that best, most stable, clusters are obtained by taking into account only low frequency descriptors. The Mahalanobis distance between points $w_i^{3;LL}$ and $w_j^{3;LL}$ is given by:

$$\delta(w_i^{3;LL}, w_j^{3;LL})^2 = (w_i^{3;LL} - w_j^{3;LL})^T \times \left(\mathcal{C}^{3;LL}\right)^{-1} \times (w_i^{3;LL} - w_j^{3;LL}) \tag{5}$$

where $\mathcal{C}^{3;LL} = \left\{ cov_{c,d}^{3;LL} \right\}$ is the covariance matrix of the points, that is:

$$cov_{c,d}^{3;LL} = \frac{1}{N} \left(\sum_{j=1}^{N} w_{c_j}^{3;LL} w_{d_j}^{3;LL} - \sum_{j=1}^{N} w_{c_j}^{3;LL} \cdot \sum_{j=1}^{N} w_{d_j}^{3;LL} \right) \tag{6}$$

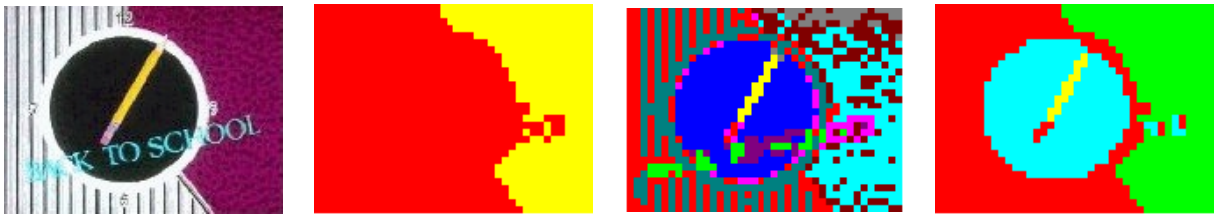
By using the Mahalanobis distance two desirable effects are obtained: First, vectors are automatically normalized (depending on the diagonal elements of the covariance matrix); second, the distance function also considers cross-correlation energies, thus texture characteristics, due to the off-diagonal elements of $\mathcal{C}^{3;LL}$.

Since different values of the k parameter can lead to different results, we iterate the k-means algorithm with $k \in [2, 10]$, and then select the "optimal" k value as the one minimizing the *validity function* V, defined as:

$$V = \frac{J'}{N \cdot \delta_{\min}^2} + \sum_{i=1}^{k'} \frac{1}{1 + |C_i|} \tag{7}$$

where k' represents the number of "good" clusters, i.e. clusters that are not too small, J' is as in Equation 4, but now it only takes into account good clusters, $\delta_{\min} = \min_{i \neq j}\{\delta(\mu_i, \mu_j)\}$ is the minimum distance between cluster centroids, and $|C_i|$ is the cardinality of cluster C_i. The first

term of Equation 7 represents the goal function J' divided by $\delta^2_{\min}$, i.e. clusters well separated provide better solutions, whereas the second term represents a penalty factor for small clusters. As an example, Figure 2 shows the results of the k-means algorithm applied to the image on the left, when $k = 2$, $k = 10$, and $k = 4$, respectively, the latter being the optimal solution according to the V validity function.



(a) (b) (c) (d)

Figure 2: (a) The input image. Clusters obtained for: (b) $k = 2$; (c) $k = 10$; (d) $k = 4$ (optimal solution). In the clustered images, points having the same color belong to the same cluster.

As a final issue, consider the particular case where the optimal solution is to have $k = 1$, which corresponds to images consisting of a uniform pattern, for which no segmentation is appropriate. Since the validity function V is not defined for $k = 1$, we resort to an analysis of the covariance matrix $\mathcal{C}^{3;LL}$. This can be geometrically represented as a 3-D ellipsoid, where each axis has a direction given by a matrix eigenvector and a length determined by its corresponding eigenvalue. Intuitively, when all the eigenvalues are small, then the wavelet coefficients have a small variance, and this can be used as an evidence that the image represents a homogeneous pattern. Since the trace of a matrix equals the sum of its eigenvalues, by just looking at the trace $\mathcal{T}_{\mathcal{C}^{3;LL}}$ of $\mathcal{C}^{3;LL}$ we can therefore deal with images for which the clustering algorithm should not be applied at all. In practice, if $\mathcal{T}_{\mathcal{C}^{3;LL}}$ is smaller than a given threshold value β, then the image is considered as a homogeneous pattern. In our tests we found that $\beta = 1000$ is an appopriate threshold value. As an example, consider the image in Figure 3 (a), whose covariance matrix is given in Figure 3 (b). It is clear that the image is a homogeneous pattern, and our perception is confirmed by the analysis of the trace of the covariance matrix whose value is $\mathcal{T}_{\mathcal{C}^{3;LL}} = 5.17 + 357.91 + 237.00 = 600.08 < 1000$.

(a)

$$\mathcal{C}^{3;LL} = \begin{pmatrix} \boxed{5.17} & 16.10 & -3.90 \\ 16.10 & \boxed{357.91} & -152.56 \\ -3.90 & -152.56 & \boxed{237.00} \end{pmatrix}$$

(b)

Figure 3: A homogeneous image (a) and its covariance matrix $\mathcal{C}^{3;LL}$ (b).

3.3 Feature Indexing

Regions obtained from the clustering phase are described using a set of *similarity features*, which are then used for image retrieval. In detail, when comparing regions, we consider information

on size and color-texture as provided by all the frequency sub-bands of the 3-rd DWT level. To this end, the similarity features for a region $R_{s,i}$ of image I_s are defined as a 37-D vector, whose components are:

Size The number of pixels in the region, $size(R_{s,i})$.

Centroid The 12-D centroid of $R_{s,i}$, $\mu_{R_{s,i}} = (\mu^{LL}_{R_{s,i}}, \mu^{LH}_{R_{s,i}}, \mu^{HL}_{R_{s,i}}, \mu^{HH}_{R_{s,i}})$, where each $\mu^{B}_{R_{s,i}}$ is a 3-D point representing the average value for each of the 3 color channels in the B sub-band.

Covariance matrices This is a 24-D vector, denoted $\mathcal{C}^{3}_{R_{s,i}}$, containing the elements of the 4 3×3 covariance matrices, $\mathcal{C}^{3;B}_{R_{s,i}}$, of the points in $R_{s,i}$. Since the covariance matrices are symmetric, only 6 values for each matrix need to be stored.

4 Image Similarity

The image similarity model of WINDSURF defines the similarity between two images as a function of the similarities among "matched" regions, as Figure 4 suggests.

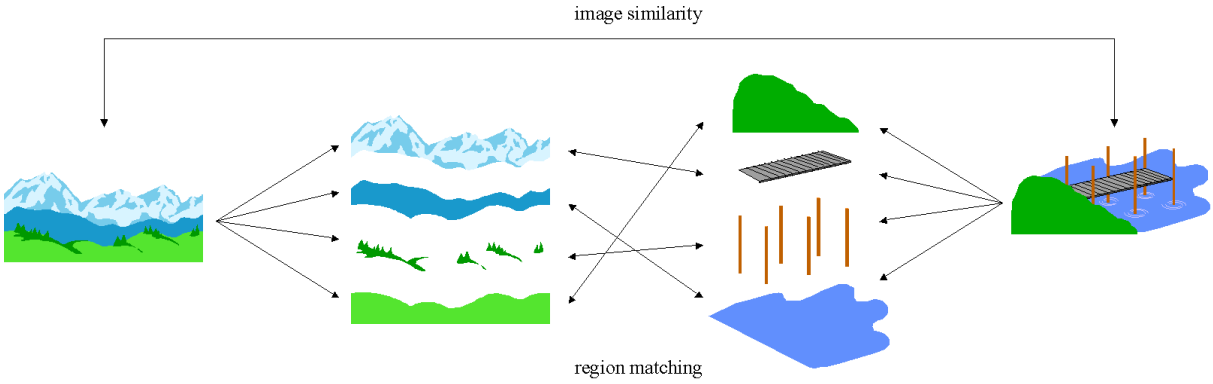


Figure 4: Similarity between images is assessed by taking into account similarity between matched regions.

To completeley characterize the image similarity model, we have therefore first to specify how similarities among regions are determined, and then how such region-based similarities are combined together.

4.1 Region Similarity

The similarity between two regions, $R_{q,i}$ (represented by the feature vector $[\mu_{R_{q,i}}, \mathcal{C}^{3}_{R_{q,i}}, size(R_{q,i})]$) of a query image I_q and $R_{s,j}$ (with feature vector $[\mu_{R_{s,j}}, \mathcal{C}^{3}_{R_{s,j}}, size(R_{s,j})]$) of a DB image I_s, is computed by WINDSURF as:

$$r_{sim}(R_{q,i}, R_{s,j}) = h(d(R_{q,i}, R_{s,j})) \tag{8}$$

where $d()$ is a distance function, and $h()$ is a so-called *correspondence function* [CPZ98] that maps distance values to similarity scores. The function $h : \Re^{+}_{0} \rightarrow [0, 1]$ has to satisfy the two

following properties:

$$\begin{aligned} h(0) &= 1 \\ d_1 \leq d_2 &\Rightarrow h(d_1) \geq h(d_2) \qquad \forall d_1, d_2 \in \Re_0^+ \end{aligned}$$

In all our experiments we use $h(d) = e^{-d/\sigma_d}$, where σ_d is the standard deviation of the distances computed over a sample of DB regions. The distance $d(R_{q,i}, R_{s,j})$ between regions $R_{q,i}$ and $R_{s,j}$ is a weighted sum, taken over the four frequency sub-bands, of the distances between color-texture descriptors, plus an additional term that takes into account the difference between the relative size of the two regions:

$$d(R_{q,i}, R_{s,j})^2 = \sum_{B \in \mathcal{B}} \gamma_B \cdot d_B(R_{q,i}, R_{s,j})^2 + \left(\frac{2}{\frac{size(R_{q,i})}{size(I_q)} + \frac{size(R_{s,j})}{size(I_s)}} \right) \cdot \left(\frac{size(R_{q,i})}{size(I_q)} - \frac{size(R_{s,j})}{size(I_s)} \right)^2 \tag{9}$$

In our experiments we equally weigh the frequency coefficients, i.e. $\gamma_B = 1 \; \forall B \in \mathcal{B}$. The second term in Equation 9 takes into account the difference in size between the regions by multiplying it by a coefficient that favors matches between large regions.

The distance $d_B(R_{q,i}, R_{s,j})$ between two regions on the frequency sub-band B is computed by using the *Bhattacharyya* metric:

$$\begin{aligned} d_B(R_{q,i}, R_{s,j})^2 = \frac{1}{2} \ln & \left(\frac{\left| \frac{\mathcal{C}^{3;B}_{R_{q,i}} + \mathcal{C}^{3;B}_{R_{s,j}}}{2} \right|}{\left| \mathcal{C}^{3;B}_{R_{q,i}} \right|^{\frac{1}{2}} \cdot \left| \mathcal{C}^{3;B}_{R_{s,j}} \right|^{\frac{1}{2}}} \right) + \\ & + \frac{1}{8} \left[\left(\mu^B_{R_{q,i}} - \mu^B_{R_{s,j}} \right)^T \times \left(\frac{\mathcal{C}^{3;B}_{R_{q,i}} + \mathcal{C}^{3;B}_{R_{s,j}}}{2} \right)^{-1} \times \left(\mu^B_{R_{q,i}} - \mu^B_{R_{s,j}} \right) \right] \end{aligned} \tag{10}$$

where $|A|$ is the determinant of matrix A. Equation 10 is composed of two terms. The second term is the Mahalanobis distance between regions centroids, where an averaged covariance matrix is used. The first term is used to compare the covariance matrices of the two regions. Note that if the two regions have the same centroid, the second term of Equation 10 vanishes, whereas the first term measures how similar the two 3-D ellipsoids are (this is the case of regions with similar colors but different texture, see Figure 5).

When computing Equation 10, we also correctly take into account those particular cases arising from singular covariance matrices. Such situations originate, for instance, from uniform images (e.g. a totally black image), where the covariance matrix is null (details are here omitted for brevity).

4.2 Combining Region-based Similarities

The basic idea of any region-based image retrieval system is that the similarity between two images depends on the similarities among component regions. What makes WINDSURF different from other systems, such as those described in Section 2, is that its similarity model can correctly define the "best matches" for a query image by taking into account *all* the information available from regions' similarities. For this we first need to define what a *matching* is.

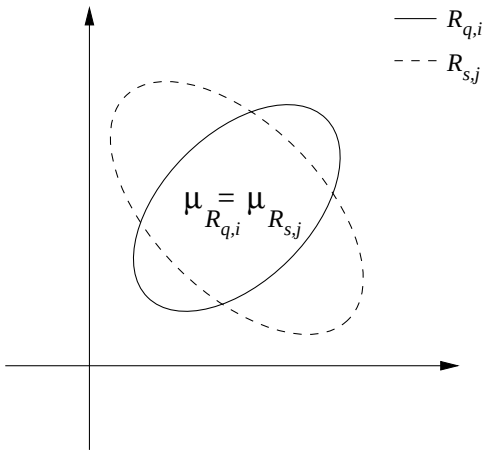


Figure 5: Two regions with different shape and equal centroids.

Definition 4.1 (Matching) *Given a query image I_q, divided into a set of regions $\mathcal{R}_q = \{R_{q,1}, \ldots, R_{q,n_q}\}$, and a DB image I_s, divided into a set of regions $\mathcal{R}_s = \{R_{s,1}, \ldots, R_{s,m_s}\}$, a* matching *between I_q and I_s is an injective function $\Gamma_s : \mathcal{R}_q \rightarrow \mathcal{R}_s \cup \{\perp\}$ that assigns to each region $R_{q,i}$ of the query image either a region of I_s or the "null match" $\perp$.*

Note that any matching satisfies, by definition, the two following constraints:

1. A region of I_q cannot match with two different regions of I_s (Figure 6 (a)).
2. Two different regions of I_q cannot match with the same region of I_s (Figure 6 (b)).

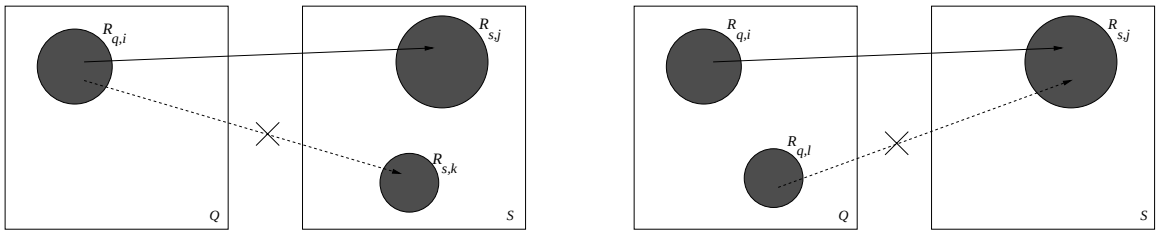


Figure 6: A region of I_q cannot match with two regions of I_s (a) and two regions of I_q cannot match with the same region of I_s (b).

Given a matching Γ_s, the corresponding similarity between I_q and I_s is computed by means of the IM_{sim} *combining function*:

$$IM_{sim}(r_{sim}(R_{q,1}, \Gamma_s(R_{q,1})), \ldots, r_{sim}(R_{q,n_q}, \Gamma_s(R_{q,n_q}))) \tag{11}$$

where it is assumed that $r_{sim}(R_{q,i}, \perp) = 0$ in case a match for $R_{q,i}$ is not defined.

The only requirement we put on the IM_{sim} function is that it has to be a monotonic non-decreasing function, i.e. $s_i \leq s'_i \implies IM_{sim}(s_1, \ldots, s_i, \ldots, s_n) \leq IM_{sim}(s_1, \ldots, s'_i, \ldots, s_n)$. This is intuitive, since better matches between regions are expected to increase the overall similarity score between corresponding images. Moreover, for the sake of simplicity, in the following we will assume that IM_{sim} is a symmetric function of its arguments.

Clearly, any different matching leads, according to Eq. 11, to a different value for the similarity of I_q and I_s. It is natural to define the "true" image similarity by only considering *optimal* matchings.

Definition 4.2 (Optimal matching) *A matching that maximizes Equation 11 is called an* optimal *matching between I_q and I_s, and will be denoted as Γ_s^{opt}.*

Definition 4.3 (Image similarity) *The similarity between I_q and I_s is defined as the value of IM_{sim} computed from an optimal matching, i.e.:*

$$\begin{aligned} I_{sim}(I_q, I_s) &= \max_{\Gamma_s}\{IM_{sim}(r_{sim}(R_{q,1}, \Gamma_s(R_{q,1})), \ldots, r_{sim}(R_{q,n_q}, \Gamma_s(R_{q,n_q})))\} \\ &= IM_{sim}(r_{sim}(R_{q,1}, \Gamma_s^{opt}(R_{q,1})), \ldots, r_{sim}(R_{q,n_q}, \Gamma_s^{opt}(R_{q,n_q}))) \end{aligned} \quad (12)$$

The following is a simple property of optimal matchings, which holds for any combining function IM_{sim}.

Property 4.1 (Maximal and complete matchings) *Let n_q be the number of regions of I_q and m_s the number of regions of I_s. If $r_{sim}(R_{q,i}, R_{s,j}) > 0$ holds for any pair of regions of I_q and I_s, then a matching Γ_s can be optimal only if it is* maximal*, that is, only if $\Gamma_s(R_{q,i})$ is undefined for exactly $\max\{n_q - m_s, 0\}$ regions of I_q. When $n_q \leq m_s$ a maximal matching is also said a* complete *matching, since for all query regions $R_{q,i}$ it is $\Gamma_s(R_{q,i}) \in \mathcal{R}_s$.*

4.3 Determining the Optimal Matching

Determining the optimal matching for images I_q and I_s can be formulated as a *generalized assignment problem*. For this, let $s_{ij} = r_{sim}(R_{q,i}, R_{s,j})$ be the similarity score between region $R_{q,i}$ of I_q and region $R_{s,j}$ of I_s, and denote with $\mathcal{H}$ the index set of pairs of matched regions, that is:

$$\mathcal{H} = \{(i,j) | \Gamma_s(R_{q,i}) = R_{s,j}\}$$

Of course, it is $|\mathcal{H}| \leq \min\{n_q, m_s\}$. Then, the goal is to maximize the function $IM_{sim}(s_{i_1 j_1}, \ldots, s_{i_{|\mathcal{H}|} j_{|\mathcal{H}|}})$, with $(i_h j_h), (i_l j_l) \in \mathcal{H}, (i_h j_h) \neq (i_l j_l)$. To this end, we introduce the variables x_{ij}, where $x_{ij} = 1$ if $\Gamma_s(R_{q,i}) = R_{s,j}$ and $x_{ij} = 0$ otherwise. Then, the generalized assignment problem is formulated as follows:

$$I_{sim}(I_q, I_s) = \max\{IM_{sim}(s_{i_1 j_1}, \ldots, s_{i_{|\mathcal{H}|} j_{|\mathcal{H}|}})\} \quad (i_h j_h), (i_l j_l) \in \mathcal{H}, (i_h j_h) \neq (i_l j_l) \quad (13)$$

$$\mathcal{H} = \{(i,j) | x_{ij} = 1\} \quad (14)$$

$$\sum_{j=1}^{m_s} x_{ij} \leq 1 \ (i = 1, \ldots, n_q) \quad (15)$$

$$\sum_{i=1}^{n_q} x_{ij} \leq 1 \ (j = 1, \ldots, m_s) \quad (16)$$

$$x_{ij} \in \{0, 1\} \ (i = 1, \ldots, n_q; \ j = 1, \ldots, m_s) \quad (17)$$

Equation 13 means that to determine the overall score $I_{sim}(I_q, I_s)$ we have to consider only the matches in $\mathcal{H}$ (Equation 14). Equation 15 (Equation 16) expresses the constraint that at most one region $R_{s,j}$ of I_s (resp. $R_{q,i}$ of I_q) can be assigned to a region $R_{q,i}$ of I_q (resp. $R_{s,j}$ of I_s).

In order to devise an algorithm to solve the generalized assignment problem, we need to consider specific choices for the IM_{sim} combining function. At present, in WINDSURF we consider the average similarity between pairs of matched regions:

$$I_{sim}(I_q, I_s) = \frac{1}{n_q} \sum_{i=1}^{n_q} r_{sim}(R_{q,i}, \Gamma_s^{opt}(R_{q,i})) = \frac{1}{n_q} \sum_{i=1}^{n_q} h(d(R_{q,i}, \Gamma_s^{opt}(R_{q,i}))) \tag{18}$$

This leads to rewrite Equation 13 as follows:

$$I_{sim}(I_q, I_s) = \frac{1}{n_q} \max \left\{ \sum_{i=1}^{n_q} \sum_{j=1}^{m_s} s_{ij} \cdot x_{ij} \right\} \tag{19}$$

The generalized assignment problem, in this case, takes the form of the well known Assignment Problem (AP), a widely studied topic in combinatorial optimization, for which the Hungarian Algorithm [Kuh55] can be used.

In case of sequential evaluation, the `ERASE` (Exact Region Assignment SEquential) algorithm, shown in Figure 7, can be used to determine the k nearest neighbors of the image query I_q within the $\mathcal{C}$ data-set. Note that `HUNG` invokes the Hungarian Algorithm on the $\{s_{ij}\}$ matrix of regions' similarity scores..

```
ERASE(I_q: query image, k: integer, C: data-set)
{ ∀ image I_s in the data-set C
  { ∀ region R_{s,j} of I_s
      ∀ region R_{q,i} of I_q compute s_ij = s(R_{q,i}, R_{s,j});
    invoke HUNG({s_ij}) obtaining, as the result, the value I_sim(I_q, I_s); }
  return the k images having the highest overall similarity scores I_sim(I_q, I_s); }
```

Figure 7: The Exact Region Assignment SEquential algorithm.

Resolution of k nearest neighbors queries by means of the `ERASE` algorithm requires the computation of similarity scores between regions in the query image and *all* the regions contained in the DB images. Algorithm complexity is, hence, linear in the database size.

To evaluate the goodness of the `ERASE` solution, we also introduce a simple heuristic method of image matching, called WINDSURF^{app}. WINDSURF^{app} first determines for each query region $R_{q,i}$ the most similar region $\Gamma_s^*(R_{q,i})$ in $\mathcal{R}_s$. Then, in case $\Gamma_s^*(R_{q,i}) = \Gamma_s^*(R_{q,i'})$ holds for two distinct query regions $R_{q,i}$ and $R_{q,i'}$, WINDSURF^{app} only keeps the best of the two assignments and discards the other one (i.e. the corresponding score is set to 0 and the query region remains unmatched).

5 Index Evaluation

In this Section we describe an index-based algorithm aiming to speed-up the evaluation of k nearest neighbors queries. This is carried out by reducing the number of *candidate images*, i.e. images for which the overall image similarity needs to be computed.

Since similarity between images is computed by combining distances between regions' features, we use a distance-based access method (DBAM), like the R*-tree [BKSS90] or the M-tree [CPZ97], to index regions extracted from the database images.[1] Such index structures are able to efficiently answer both range and k nearest neighbors queries, as well as to perform a *sorted access* to the data, i.e. they can output regions one by one in increasing order of distance with respect to a query region [HS95].

In order to deal with "compound" queries, where multiple query regions are specified, a query processing algorithm based on multiple sorted access index scans is needed. To retrieve the best matches for the query regions, we run a sorted access to the indexed regions for each region in the query image. Clearly, the problem is to devise a suitable condition to stop such *sorted access phase* so that we are guaranteed that the k best images can be correctly determined without looking at the whole data-set. More precisely, the stop condition has to guarantee that the k nearest neighbor images of the query image I_q are among the so-called *candidate images*, i.e. those images for which at least one region has been retrieved during the sorted access phase (Figure 8).

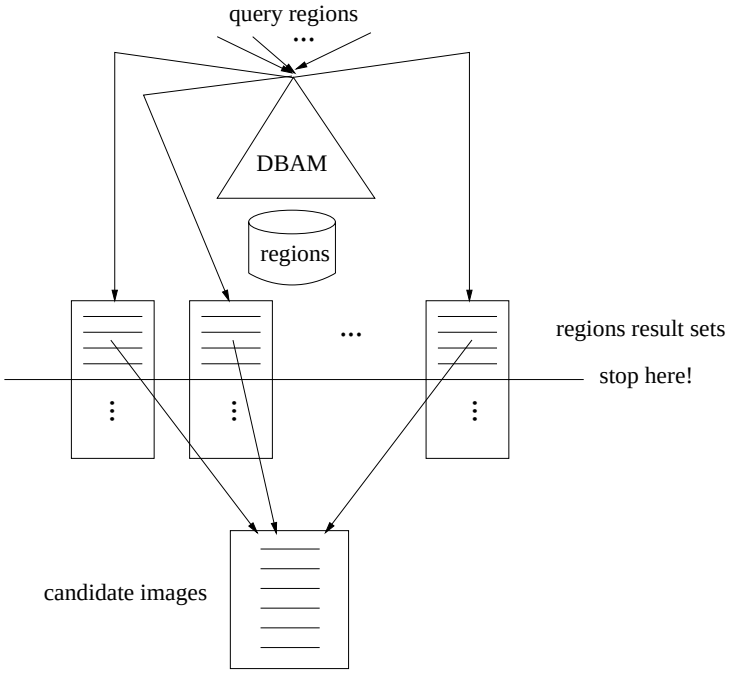


Figure 8: Producing the candidate set of images from the sorted access phase.

A first naïve approach to resolve compound queries with DBAMs goes as follows. For each region $R_{q,i}$ of the query image I_q, we execute a k nearest neighbors query, that is, we determine the k regions in the data-set most similar to $R_{q,i}$. Then, we compute an optimal matching for

[1] In WINDSURF we use the M-tree index [CPZ97], but other choices are possible.

all the images for which at least one region has been returned by the previous step. Note that this is, indeed, the query processing approach used by Blobworld.

This algorithm guarantees that the number of candidate images is not higher than $n_q \cdot k$. Such a solution is indeed quite efficient, but it is *not* correct. As an example, consider the case where $n_q = 2$, $k = 1$, and assume that the regions' similarity scores obtained by the two sorted access scans are as in Table 1.

$R_{q,1}$			$R_{q,2}$		
region	image	similarity	region	image	similarity
$R_{1,1}$	I_1	0.90	$R_{3,2}$	I_3	0.87
$R_{2,2}$	I_2	0.85	$R_{2,1}$	I_2	0.79
$R_{4,1}$	I_4	0.83	$R_{3,3}$	I_3	0.75
$R_{3,3}$	I_3	0.71	$R_{1,1}$	I_1	0.72
$R_{2,1}$	I_2	0.69	$R_{1,2}$	I_1	0.70
⋮	⋮	⋮	⋮	⋮	⋮

Table 1: A sorted access example for a query image with two regions $R_{q,1}$ and $R_{q,2}$.

It is plain to see that the image most similar to I_q is the image I_2 (the overall similarity score, computed as the average sum of regions similarities, is $(0.9 + 0.7)/2 = 0.80$ for image I_1, $(0.85 + 0.79)/2 = 0.82$ for I_2, $(0.71 + 0.87)2 = 0.79$ for I_3, with other images leading to lower scores), whereas the candidate set only contains images I_1 and I_3.

In order to find a correct condition to stop the sorted accesses, we start from Fagin's $\mathcal{A}_0$ algorithm [Fag96]. The $\mathcal{A}_0$ algorithm stops the sorted access phase when at least k objects are included in all the index scans results. The only requirement for the $\mathcal{A}_0$ algorithm is that the function applied to combine objects' scores (in our case, the IM_{sim} function) has to be monotonic. Applying the $\mathcal{A}_0$ algorithm to the optimal image matching problem would be as in Figure 9.

$\mathcal{A}_0$(`$I_q$: query image, $k$: integer, $\mathcal{T}$: DBAM`)
{ ∀ `region` $R_{q,i}$ `of` I_q`, open a` *`sorted access`* `index scan on` $\mathcal{T}$ `and insert images containing result regions in the set` X^i`;`
`stop the sorted accesses when there are at least` k `images in the intersection` $L = \cap_i X^i$`;`
`for each image` I_s `in the` *`candidate set`* $\cup_i X^i$`, compute the optimal assignment;` (*`random access`*)
`return the` k `images having the highest overall similarity scores` $I_{sim}(I_s, I_q)$`;` }

Figure 9: The $\mathcal{A}_0$ algorithm for the optimal image matching problem.

$\mathcal{A}_0$, however, does not guarantee yet that the k best images are included in the candidate set, since its stopping condition does not take into account that assignment of regions has to be a matching. Just consider, as an example, the case depicted in Table 2, where $n_q = 2$ and $k = 1$. Here, as opposed to the case of Table 1, it is not correct to stop the sorted access phase

at the second step, since image I_2 has been found for both query regions with the same region $R_{2,1}$; therefore, we cannot find a matching for image I_2 by using only regions that have been seen during the sorted access phase.

$R_{q,1}$			$R_{q,2}$		
region	image	similarity	region	image	similarity
$R_{1,1}$	I_1	0.90	$R_{3,2}$	I_3	0.87
$R_{2,1}$	I_2	0.85	$R_{2,1}$	I_2	0.79
$R_{4,1}$	I_4	0.83	$R_{3,3}$	I_3	0.75
$R_{3,3}$	I_3	0.71	$R_{1,1}$	I_1	0.72
$R_{2,3}$	I_2	0.69	$R_{1,2}$	I_1	0.70
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Table 2: Another sorted access example for a query image with two regions $R_{q,1}$ and $R_{q,2}$.

To ensure that the k best results are included into the set of candidate images, the stopping condition of $\mathcal{A}_0$ algorithm has to be modified to test correctness of regions' assignments (see Definition 4.1). The sorted access phase can be stopped as soon as a complete matching (Property 4.1) is found, by taking into account only regions returned by index scans.[2] In the example of Table 2, hence, we stop the sorted access phase after the fourth step, since image I_3 has a complete matching ($\Gamma_3(R_{q,1}) = R_{3,3}$ and $\Gamma_3(R_{q,2}) = R_{3,2}$). It should be noted, however, that this is *not* the best result for I_q (image I_1 leads to the best overall score of 0.8). In other words, the sorted accesses can be stopped as soon as it is guaranteed that each image outside of the candidate set leads to an overall similarity score lower than that of the k-th best image, i.e. when optimal matchings for non-candidate images could only lead to lower scores with respect to the k-th best matching within the candidate set.

Consider again, as an example, the case where $n_q = 2$ and $k = 1$, and refer to Table 2. After the first step, the candidate set is $\{I_1, I_3\}$ with overall scores $(0.9 + 0)/2 = 0.45$ and $(0+0.87)/2 = 0.435$, respectively. Since an image outside the candidate set could potentially lead to an overall score of $(0.9 + 0.87)/2 = 0.885$, we have to continue the sorted access phase. After the second step, we add image I_2 to the candidate set with an overall score of $(0.85+0)/2 = 0.425$ (remember that region $R_{2,1}$ can match at most one region of I_q); therefore, the sorted accesses cannot be stopped yet. At the third step, also image I_4 is added to the candidate set, with a score of $(0.83 + 0)/2 = 0.415$. Finally, at fourth step, we obtain a complete matching for image I_3 ($\Gamma_3(R_{q,1}) = R_{3,3}$ and $\Gamma_3(R_{q,2}) = R_{3,2}$) with a score of $(0.71 + 0.87)/2 = 0.79$. In this case, the sorted access phase can be stopped, since images outside of the candidate set can only lead to lower scores (at most $(0.71 + 0.72)/2 = 0.715$). The monotonicity of the combining function IM_{sim} is used here to ensure algorithm correctness (see below). Note, however, that image I_3 is *not* the best result for I_q, since image I_1 leads to the best overall score of 0.8. In order to solve the optimal image matching problem on the set of candidate images, we need to compute similarity scores between query regions and *all* the regions of candidate images.

From above example, it is clear that the sorted access phase can be stopped as soon as a

[2]By the way, this is the reason why Blobworld algorithm is not correct, since its stopping condition cannot guarantee the existence of a complete matching.

complete matching is found, by taking into account only regions returned by index scans. This leads to the $\mathcal{A}_0^{WS}$ algorithm shown in Figure 10.

```
A_0^WS(I_q: query image, k: integer, T: DBAM)
{ ∀ region R_{q,i} of I_q, open a sorted access index scan
    on T and insert result regions in the set X^i;
  stop the sorted accesses when there are at least k images for which a complete
    matching exists, considering only regions in ∪_i X^i;
  ∀ image I_s having regions in ∪_i X^i,
    ∀ pair R_{q,i}, R_{s,j}
      if R_{s,j} ∉ X^i compute score s_ij; (random access)
    compute the optimal assignment; (combining phase)
  return the k images having the highest overall similarity scores I_sim(I_q, I_s); }
```

Figure 10: The $\mathcal{A}_0^{WS}$ algorithm.

The *random access* phase consists in computing those similarity scores s_{ij} between query regions and regions of candidate images not present in the X^i regions result sets. After that, the *combining phase* determines the optimal matchings for all the candidate images.

Correctness of the $\mathcal{A}_0^{WS}$ algorithm follows from the monotonicity of the IM_{sim} combining function. Without loss of generality, consider the case $k = 1$ and assume by contradiction that the nearest neighbor image, say I_{nn}, of I_q is *not* in the candidate set. Also observe that the candidate set includes (at least) one image, say I_s, for which a complete matching has been obtained. We prove that $I_{sim}(I_q, I_s) \geq I_{sim}(I_q, I_{nn})$. Because of the monotonicity of IM_{sim} it is enough to show that, for each $i \in [1, n_q]$, it is

$$r_{sim}(R_{q,i}, \Gamma_s(R_{q,i})) \geq r_{sim}(R_{q,i}, \Gamma_{nn}^{opt}(R_{q,i}))$$

where Γ_s is the matching obtained for I_s from the sorted access phase, and Γ_{nn}^{opt} is the optimal matching for I_{nn}. Assume that there exists a value of i for which it is:

$$r_{sim}(R_{q,i}, \Gamma_{nn}^{opt}(R_{q,i})) > r_{sim}(R_{q,i}, \Gamma_s(R_{q,i}))$$

However, this is impossible since the region $\Gamma_{nn}^{opt}(R_{q,i})$ of I_{nn} does not belong, by hypothesis, to the set X^i obtained from the i-th sorted access scan. This is enough to prove that $I_{sim}(I_q, I_s) \geq I_{sim}(I_q, I_{nn})$.

The index evaluation of compound queries, thus, will have a twofold impact on query evaluation: First, the use of an index can reduce the number of distance computations needed for assessing image similarity; second, the number of images on which the Hungarian algorithm has to be run is reduced by considering only images in the candidate set.

6 Experimental results

Preliminary experimentation of the proposed techniques has been performed on a sample medium-size data-set consisting of about 2,000 real-life images, yielding more than 10,000 regions, ex-

tracted from a CD-ROM from *IMSI-PHOTOS*.[3] The query workload consists of about 100 randomly chosen images not included in the data-set. All experiments were performed on a Pentium II 450 MHz PC equipped with 64MB of main memory and running Windows NT 4.0.

6.1 Efficiency

The first set of experiments we present concerns the efficiency of the proposed approach. In order to test the performance of the $\mathcal{A}_0^{WS}$ index-based algorithm, in Figure 11 we compare the number of candidate images, i.e. the images on which the Hungarian algorithm has to be applied, as a function of the number of query regions.[4] Of course, for the `ERASE` algorithm the number of candidate images equals the number of images in the data-set, whereas for the index version this number depends both on values of k and of the number of query regions. As the graph shows, the $\mathcal{A}_0^{WS}$ algorithm is indeed very efficient in reducing the number of candidate images, even if its performance deteriorates as the number of query regions increases. This is intuitive, since the complexity of finding k objects in the intersection of n_q sets augments with n_q.

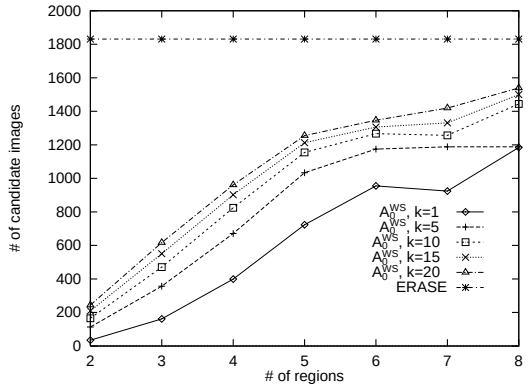


Figure 11: Average number of candidate images as a function of the number of query regions.

Another element affecting performance is the number of computed distances between regions. In Figure 12 (a) we show the number of computed distances for the `ERASE` and the $\mathcal{A}_0^{WS}$ algorithms, as a function of k, with a number of query regions $n_q = 3$. In order to reduce the number of distances to be computed for the index-based algorithm, we also considered an approximate version of the $\mathcal{A}_0^{WS}$ algorithm, called $\mathcal{A}_0^{WS_{app}}$. In this case, the random access phase computes the optimal matching for each candidate image by taking into account only regions returned by the sorted access phase, i.e. no new distance is computed. Average number of distance computations for the $\mathcal{A}_0^{WS_{app}}$ algorithm is also shown in Figure 12 (a). The graph shows that the index-based approach is not very efficient in reducing the number of computed distances. We believe that this is due to the low cardinality of the data-set: Increasing the number of images in the data-set would have a beneficial effect on the performance of index-based algorithms (whose search costs grow logarithmically with the number of indexed objects) with respect to that of sequential ones.

[3] IMSI MasterPhotos 50,000: `http://www.imsisoft.com`.

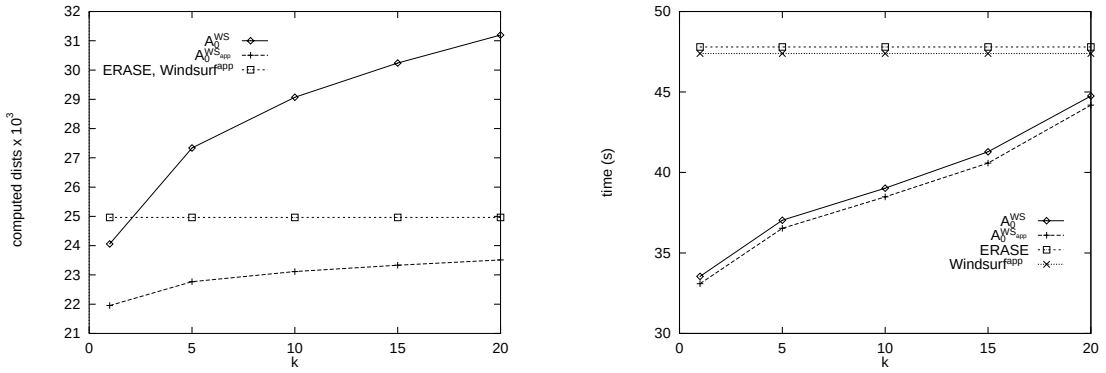
[4] Unless otherwise specified, all the graphs presented here show numbers averaged over all the images contained in the query workload.

Finally, in Figure 12 (b) we compare query response times as a function of k (with a constant value of $n_q = 3$). The graph shows average query evaluation times (in seconds) for the ERASE algorithm, the WINDSURF^{app} heuristic matching algorithm, and the two index-based algorithms, $\mathcal{A}_0^{WS}$ and $\mathcal{A}_0^{WS_{app}}$, respectively. From the graph it can be deduced that:

(i) The lower complexity of image matching for the WINDSURF^{app} algorithm with respect to the ERASE algorithm does not pay off in reducing query evaluation times. This is due to the fact that, if n_q is low (as it is in our case), finding the optimal result is very easy.

(ii) The index-based algorithms really succeed in cutting down query resolution times, even if difference in performance reduces with increasing values of k.

(iii) The approximate $\mathcal{A}_0^{WS_{app}}$ algorithm has performance similar to that of the exact $\mathcal{A}_0^{WS}$ algorithm. This demonstrates that the performance improvement with respect to sequential query evaluation is due to the lower number of candidate images, and that the number of computed distances has a minor impact on performance.



(a) (b)

Figure 12: Average number of computed distances (a) and average query resolution time (b) as a function of k ($n_q = 3$).

6.2 Effectiveness

In order to compare the "goodness" of results obtained by approximate algorithms (i.e. the WINDSURF^{app} heuristic matching algorithm and $\mathcal{A}_0^{WS_{app}}$) with respect to those obtained by exact ones (ERASE and $\mathcal{A}_0^{WS}$), we need a performance measure able to compare results of k nearest neighbors queries. Such measure should compare two sorted lists of results, that is, it should contrast *ranks* (positions) of images in exact and approximate results. Given the i-th image in the approximate result, its rank, rank(i), is given by the position of that image in the exact result. As an example, consider the case when $k = 1$. The "goodness" of an approximate result with respect to the exact one can be obtained by just taking into account rank(1): The lower rank(1) is, the better the approximation works. This measure can be easily extended to the case where $k > 1$ by considering the ranks of all the k images in the approximate result, i.e. rank(1), ... , rank(k).

In [WB00], the *normalized rank sum* (nrs) is used to quantify the loss of result quality when k nearest neighbors queries are approximately evaluated. The nrs is defined as:

$$nrs = \frac{k(k+1)}{2 \cdot \sum_{i=1}^{k} \text{rank}(i)} \tag{20}$$

The nrs is computed as the inverse of the sum of all the ranks of the images in the approximate result. Thus, higher values of nrs are to be preferred. This measure, however, is not able to capture inversions in the result (e.g. when image I_s is ranked higher than image $I_{s'}$ in the approximate result and lower in the exact result), since no difference between ranks of images in the approximate and in the exact results is taken into account.

In [ZSAR98], the *precision of approximation* measure P is introduced, which is defined as:

$$P = \frac{1}{k} \sum_{i=1}^{k} \frac{i}{\text{rank}(i)} \tag{21}$$

P, therefore, measures the relative error in ranking for all the images in the approximate result. This measure, however, relies on the assumption that $i \leq \text{rank}(i)$, thus no inversions on results are allowed.

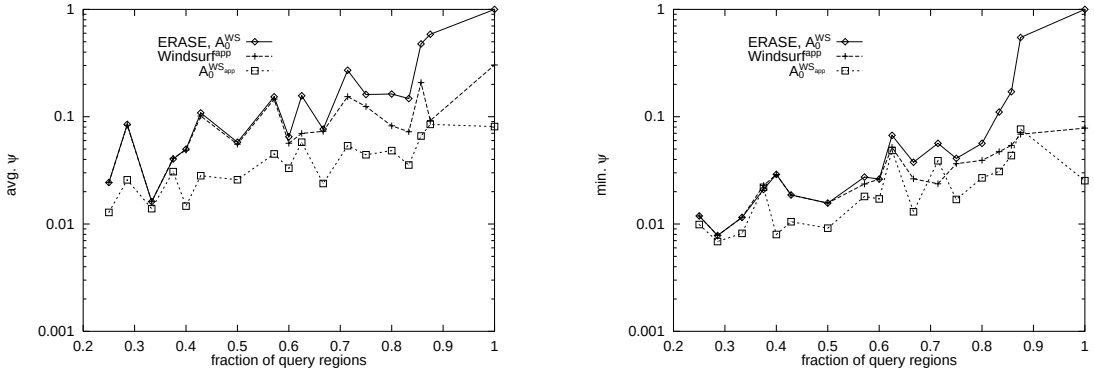
To overcome above limitations in quality measures, we introduce a new measure, the normalized rank difference sum ψ. To compute ψ, we sum differences in rankings for images in the approximate result and normalize by k. Normalization of the measure in the interval $[0, 1]$ leads to the formulation of ψ as follows:

$$\psi = \frac{1}{1 + \frac{1}{k} \left(\sum_{i=1}^{k} \mathbf{1}(\text{rank}(i) - i)^p \right)^{1/p}} \tag{22}$$

where $\mathbf{1}()$ is the ramp function ($\mathbf{1}(x) = 0$ if $x < 0$, $\mathbf{1}(x) = x$ if $x \geq 0$), and p is an integer parameter (we used $p = 2$ in our experiments). Values of ψ close to 1 indicate high quality of the approximate result. The use of the ramp function $\mathbf{1}()$ is needed to avoid counting twice the effects of inversions in ranking. For instance, consider the case where $k = 3$ and the exact result is I_1, I_2, and I_3. If the approximate result is I_1, I_3, I_2, it is $rank(1) = 1$, $rank(2) = 3$, and $rank(3) = 2$. Accordingly to Equation 22, and setting $p = 2$, the value of ψ is computed as:

$$\psi = \frac{1}{1 + \frac{1}{3} \left(\mathbf{1}(1-1)^2 + \mathbf{1}(3-2)^2 + \mathbf{1}(2-3)^2 \right)^{1/2}} = \frac{1}{1 + \frac{1}{3} \left(0 + 1 + 0 \right)^{1/2}} = 0.75$$

Figure 13 shows average (a) and minimum (b) values of ψ for exact and approximate algorithms as a function of the fraction of query regions used to query the database (the value of k is kept fixed at 20, other values lead to similar results and are omitted here for brevity). This is to show the effectiveness of different approaches when only some regions of the query image are used for the query (this can be done in order to reduce the query response time or just because we are interested only in some objects included in the query image). Both graphs exhibit similar trends: The effectiveness of the $\mathcal{A}_0^{WS_{app}}$ algorithm is almost always the lowest, and, for all curves, ψ only reaches high values when the fraction of query regions is close to 1. Figure 13 (b) shows that, in order to find a "good" result, we have to use *all* the regions in the query image. From Figure 13 (a), on the other hand, we see that approximate algorithms lead to a low effectiveness, even if, as we have seen before, they attain slightly better performance with respect to their exact counterparts.



(a) (b)

Figure 13: Average (a) and minimum (b) ψ as a function of the fraction of query regions ($k = 20$).

6.3 Comparison with other CBIR techniques

In order to evaluate the effectiveness of WINDSURF, we compare the results of WINDSURF for a number of image queries with those obtained by applying the method proposed in [SO95], denoted SO.

From a semantic point of view, results obtained by WINDSURF are considerably better with respect to those obtained by the SO method. As an example, consider Figure 14: Results for SO (SO1 - SO5) contain images semantically uncorrelated to the query image (e.g. image (SO3), a house, and image (SO5), a harbour). As for the results of WINDSURF (WS1 - WS5), all of them present a "sky" region and a darker area.

The superior effectiveness of our approach is confirmed when considering "difficult" queries, i.e. queries having a low number of similar images in the DB. In Figure 15 we show the results for a query having only two similar images: For SO, none of the two images is included in the result. WINDSURF, on the other hand, retrieves both images.

Finally, we compared the two approaches when dealing with "partial-match" queries, i.e. queries specifying only a part of the image. As an example, consider Figure 16, where the query image is obtained by "cropping" a DB image, namely, the dome of St. Peter in Rome. With WINDSURF all the retrieved images refer to St. Peter, with the only exception of image (WS3), representing the dome of St. Marcus in Venice. Indeed, the query image was extracted from image (WS1). When we anaylze the result obtained by using SO, we see that only one image related to the query image is retrieved in third position, whereas other images, with the exception of image (SO2) (again the dome of St. Marcus), are totally uncorrelated to the query image.

7 Conclusions

In this work we have presented WINDSURF, a region-based image retrieval system. WINDSURF uses a wavelet-based clustering approach to segment images into homogeneous regions, and then applies a *correct* image matching algorithm to retrieve the k nearest neighbor images of a query image. Similarity between images is assessed by means of a distance function comparing region features and combining the results at a global level. Towards this goal, both a sequential

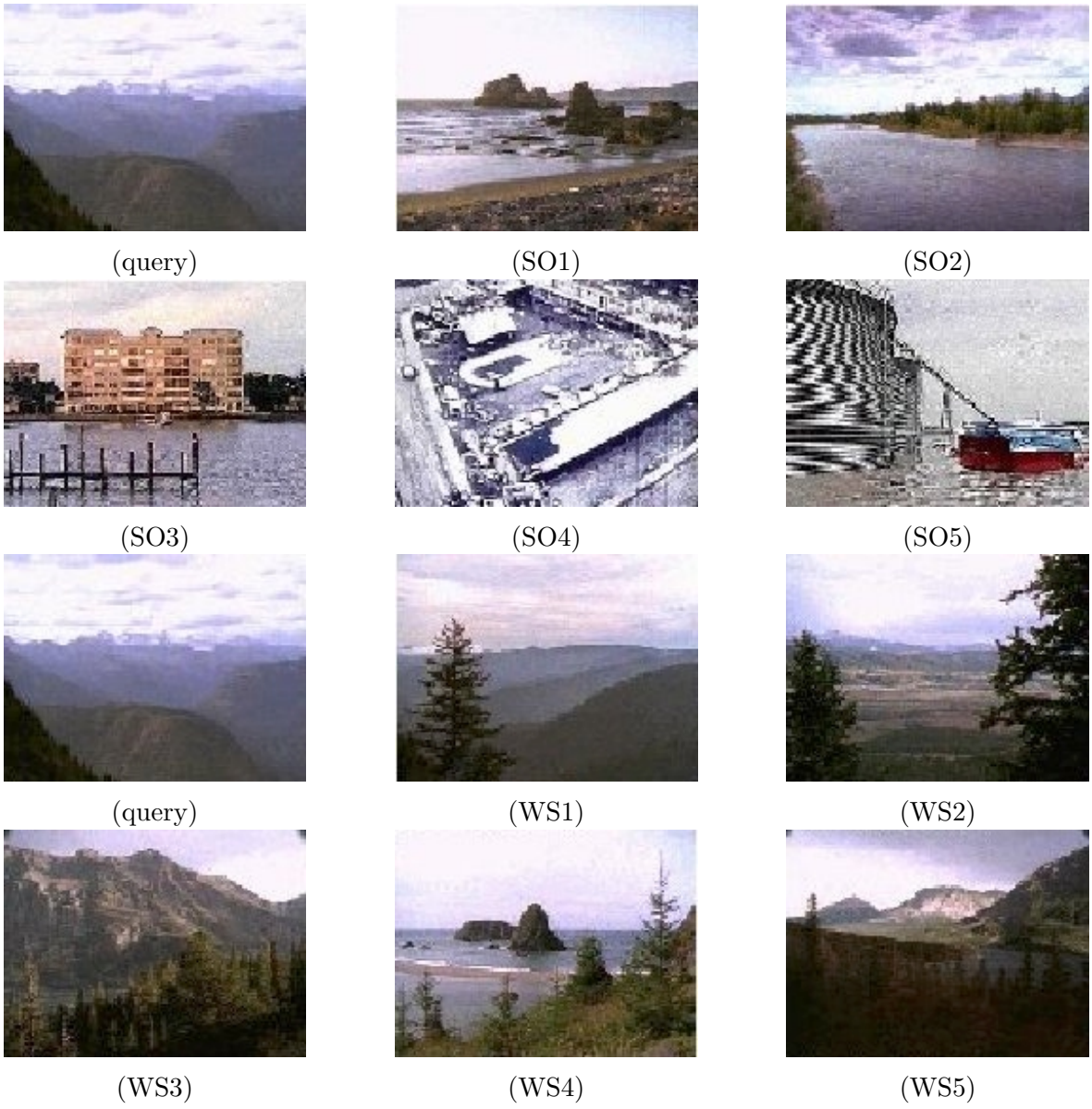


Figure 14: Results for the "mountains" query.

(`ERASE`) and an index-based ($\mathcal{A}_0^{WS}$) algorithms have been described. Preliminary experimental results demonstrate the superior retrieval effectiveness of WINDSURF with respect to the method described in [SO95] and show that our image matching algorithm is very effective with respect to existing heuristic approaches. However, from the efficiency point of view, there is still room for improvement. In particular, we have observed that the $\mathcal{A}_0^{WS}$ algorithm, even if more efficient as compared to the sequential one, is not successful in reducing the number of distance computations needed to answer a query. In order to overcome this limitation, we plan to employ approximate techniques for index access [CP00]. Another issue that needs to be investigated regards the possible parallelization of multiple index scans, along the lines described in [BEKS00].

(query) (SO1) (SO2)

(query) (WS1) (WS2)

Figure 15: Results for the "bridge" query.

References

[BCGM98] Serge Belongie, Chad Carson, Hayit Greenspan, and Jitendra Malik. Color- and texture-based image segmentation using EM and its application to content-based image retrieval. In *Proceedings of the 6th International Conference on Computer Vision ICCV'98*, Mumbai, India, January 1998.

[BDV99] Stefano Berretti, Alberto Del Bimbo, and Enrico Vicario. Managing the complexity of match in retrieval by spatial arrangement. In *International Conference on Image Analysis and Processing (ICIAP'99)*, Venezia, Italy, September 1999.

[BEKS00] Bernhard Braunmüller, Martin Ester, Hans-Peter Kriegel, and Jörg Sander. Efficiently supporting multiple similarity queries for mining in metric databases. In *Proceedings of the 16th International Conference on Data Engineering (ICDE 2000)*, pages 256–267, San Diego, CA, February 2000.

[BKSS90] Norbert Beckmann, Hans-Peter Kriegel, Ralf Schneider, and Bernhard Seeger. The R*-tree: An efficient and robust access method for points and rectangles. In *Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data*, pages 322–331, Atlantic City, NJ, May 1990.

[CK93] T. Chang and C.-C.J. Kuo. Texture analysis and classification with tree-structured wavelet transform. *IEEE Transactions on Image Processing*, 2(4):429–440, October 1993.

[CP00] Paolo Ciaccia and Marco Patella. PAC nearest neighbor queries: Approximate and controlled search in high-dimensional and metric spaces. In *Proceedings of the 16th International Conference on Data Engineering (ICDE 2000)*, pages 244–255, San Diego, CA, February 2000.

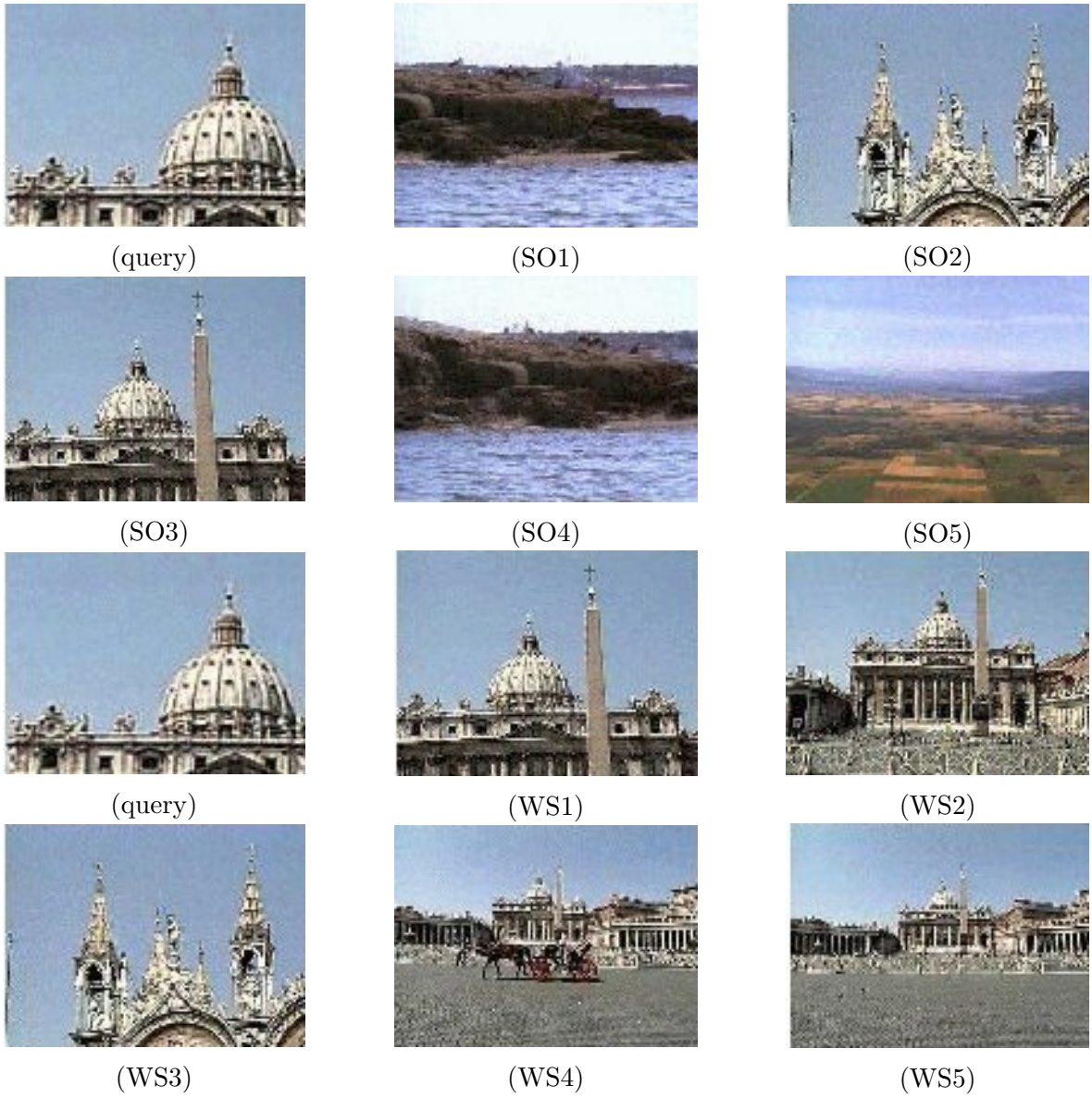


Figure 16: Results for the "dome" query.

[CPZ97] Paolo Ciaccia, Marco Patella, and Pavel Zezula. M-tree: An efficient access method for similarity search in metric spaces. In *Proceedings of the 23rd International Conference on Very Large Data Bases (VLDB'97)*, pages 426–435, Athens, Greece, August 1997.

[CPZ98] Paolo Ciaccia, Marco Patella, and Pavel Zezula. Processing complex similarity queries with distance-based access methods. In *Proceedings of the 6th International Conference on Extending Database Technology (EDBT'98)*, pages 9–23, Valencia, Spain, March 1998.

[CTB+99] Chad Carson, Megan Thomas, Serge Belongie, Joseph M. Hellerstein, and Jitendra Malik. Blobworld: A system for region-based image indexing and retrieval. In

Proceedings of the 3rd International Conference on Visual Information Systems VISUAL'99, pages 509–516, Amsterdam, The Netherlands, June 1999.

[Dau92] Ingrid Daubechies. *Ten Lectures on Wavelets*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1992.

[Fag96] Ronald Fagin. Combining fuzzy information from multiple systems. In *Proceedings of the 15th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'96)*, pages 216–226, Montreal, Canada, June 1996.

[Fal96] Christos Faloutsos. *Searching Multimedia Database by Content*. Kluwer Academic Publishers, 1996.

[FEF+94] Christos Faloutsos, Will Equitz, Myron Flickner, Wayne Niblack, Dragutin Petkovic, and Ron Barber. Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, 3(3/4):231–262, July 1994.

[GR95] Venkat N. Gudivada and Vijay V. Raghavan. Content-based image retrieval systems. *IEEE Computer*, 28(9):18–22, September 1995. Guest Editors' Introduction.

[HS95] Gisli R. Hjaltason and Hanan Samet. Ranking in spatial databases. In *Proceedings of the 4st International Symposium on Advances in Spatial Databases (SSD'95)*, pages 83–95, Portland, ME, August 1995.

[Kuh55] Harold W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistic Quarterly*, 2:83–97, 1955.

[NRS99] Apostol Natsev, Rajeev Rastogi, and Kyuseok Shim. WALRUS: A similarity retrieval algorithm for image databases. In *Proceedings 1999 ACM SIGMOD International Conference on Management of Data*, Philadelphia, PA, June 1999.

[OS95] Virginia E. Ogle and Michael Stonebraker. Chabot: Retrieval from a relational database of images. *IEEE Computer*, 28(9):40–48, September 1995.

[PPS96] Alex Pentland, Rosalind W. Picard, and Stan Sclaroff. Photobook: Content-based manipulation of image databases. In Borko Furht, editor, *Multimedia Tools and Applications*, chapter 2, pages 43–80. Kluwer Academic Publishers, 1996.

[Sal88] Gerard Salton. *Automatic Text Processing: The Transformational, Analysis, and Retrieval of Information by Computer*. Addison-Wesley, Reading, MA, 1988.

[SC96] John R. Smith and Shih-Fu Chang. VisualSEEk: A fully automated content-based image query system. In *Proceedings of the 4th ACM International Conference on Multimedia*, pages 87–98, Boston, MA, November 1996. `http://www.ctr.columbia.edu/visualseek/`.

[Smi97] John R. Smith. *Integrated Spatial and Feature Image Systems: Retrieval, Analysis and Compression*. PhD thesis, Columbia University, 1997.

[SO95] Markus Stricker and Markus Orengo. Similarity of color images. In *Storage and Retrieval for Image and Video Databases SPIE*, volume 2420, pages 381–392, San Jose, CA, February 1995.

[TCH00] Megan Thomas, Chad Carson, and Joseph M. Hellerstein. Creating a customized access method for Blobworld. In *Proceedings of the 16th International Conference on Data Engineering ICDE 2000*, page 82, San Diego, CA, March 2000.

[UVJ+97] Geert Uytterhoeven, Filip Van Wulpen, Maarten Jansen, Dirk Roose, and Adhemar Bultheel. WAILI: Wavelets with integer lifting. Technical Report 262, Department of Computer Science, Katholieke Universiteit Leuven, Heverlee, Belgium, July 1997.

[VSLV] Gert Van de Wouver, Paul Scheunders, Stefan Livens, and Dirk Van Dyck. Color texture classification by wavelet energy-correlation signatures. To appear on *Pattern Recognition*.

[WB00] Roger Weber and Klemens Böhm. Trading quality for time with nearest-neighbor search. In *Proceedings of the 7th International Conference on Extending Database Technology (EDBT2000)*, Konstanz, Germany, March 2000.

[WWFW97] James Ze Wang, Gio Wiederhold, Oscar Firschein, and Sha Xin Wei. Wavelet-based image indexing techniques with partial sketch retrieval capability. In *Proceedings of the 4th IEEE Forum on Research and Technology Advances in Digital Libraries (ADL'97)*, Washington, DC, May 1997.

[XB91] Xuanli Lisa Xie and Gerardo Beni. A validity measure for fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(8):841–847, August 1991.

[ZSAR98] Pavel Zezula, Pasquale Savino, Giuseppe Amato, and Fausto Rabitti. Approximate similarity retrieval with M-trees. *The VLDB Journal*, 7(4):275–293, 1998.

A The Wavelet Transform

The basic idea of the wavelet transform (WT) is similar to that of Fourier transform: Approximate a signal through a set of basic mathematical functions. However, wavelet functions are able to give a multi-resolution representation of the signal, since each frequency component can be analyzed with a different resolution and scale, whereas the Fourier transform divides the time-frequency domain in a homogeneous way. This allows the WT to represent discontinuities in the signal by using "short" functions and, at the same time, to emphasize low frequency components using "wide" functions.

The Continuous WT decomposes a 1-D signal $f(x)$ into a set of *scaling functions* by using a set of wavelet basis functions $\{\psi_{a,b}\}$:

$$(W_a f)(b) = \int f(x)\psi^*_{a,b}(x)\mathrm{d}x \tag{23}$$

where each wavelet basis function is obtained from a *mother wavelet* $\psi(x)$ by scaling and shifting:

$$\psi_{a,b}(x) = \frac{1}{\sqrt{a}}\psi\left(\frac{x-b}{a}\right) \tag{24}$$

The mother wavelet should only satisfy the zero-average condition, i.e. $\int \psi(x)\mathrm{d}x = 0$.

The Discrete WT is obtained by taking $a = 2^n$ and $b \in \mathbb{Z}$. The oldest, and simplest, example of a mother wavelet is the Haar function, which was first introduced in 1910, and is composed by a pair of rectangular pulses:

$$\psi(x) = \begin{cases} 1 & 0 \leq x < 1/2 \\ -1 & 1/2 \leq x < 1 \\ 0 & \text{otherwise} \end{cases} \tag{25}$$

The following examples show how a discrete signal is decomposed by means of the Haar wavelet.

Example A.1 *Consider a discrete signal $x = (x_0, x_1, \ldots, x_{2^L-1})$ having length 2^L. The DWT is computed through the following steps:*

1. *For each pair of consecutive samples (x_{2i}, x_{2i+1}), $(0 \leq i < 2^{L-1})$, compute $a_i^1 = \frac{1}{\sqrt{2}}(x_{2i} + x_{2i+1})$ and $d_i^1 = \frac{1}{\sqrt{2}}(x_{2i} - x_{2i+1})$.*

2. *Consider the new signal $(a_0^1, \ldots, a_{2^{L-1}-1}^1)$ and proceed as in step 1., obtaining a_i^2 and d_i^2 $(0 \leq i < 2^{L-2})$.*

3. *Continue until a single value of a_0^L is obtained.*

The Haar transform of x is given by the set of "difference" values d_i^l $(0 < l \leq L,\ 0 < i < 2^{l-1})$, and the "average" value for the last level a_0^L. In the frequency domain, the values a_i^l correspond to the output of a low pass filter*, thus representing low-frequency information, whereas the d_i^l values correspond to the output of a* high pass filter*, thus representing high-frequency information.*

In our case, the signal is a 2-D color image, where the "time" domain is the spatial location of pixels and the frequency domain is the color variation between adjacent pixels. In order to build an orthonormal wavelet basis for the 2-dimensional space, one can start from the 1-dimensional domain and compute the product of two 1-dimensional basis functions, that is, $\Psi_{j_1,k_1;j_2,k_2}(x_1, x_2) = \psi_{j_1,k_1}(x_1) \cdot \psi_{j_2,k_2}(x_2)$. If the image to be processed has dimension $N \times M$, the first transformation step decomposes the signal into four sub-images of dimension $N/2 \times M/2$, representing the sub-bands in the frequency domain. The obtained sub-images are labelled as LL, LH, HL, HH, where L and H represent low- and high-frequency information, respectively, and the first position refers to the horizontal direction, whereas the second position refers to the vertical direction:

LL: Low-frequency information in both the horizontal and vertical directions.

LH: Low-frequency information in the horizontal direction, high-frequency information in the vertical direction.

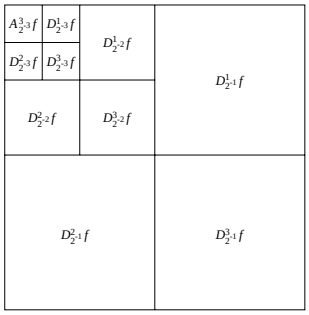


Figure 17: The sub-images $D^k_{2^{-j}}f, A^d_{2^{-L}}f$ in the "wavelet" image representation.

HL: High-frequency information in the horizontal direction, low-frequency information in the vertical direction.

HH: High-frequency information in both the horizontal and vertical directions.

The second transformation level decomposes the LL sub-image, obtaining four images of dimension $N/4 \times M/4$, and so on. Figure 17 shows the decomposition of the frequency domain at different scale levels: $A^d_{2^{-L}}f$ contains low-frequency information, whereas $D^1_{2^{-j}}f$, $D^2_{2^{-j}}f$, and $D^3_{2^{-j}}f$ contain horizontal, vertical and diagonal information, respectively.