A Sound Algorithm for Region-Based Image Retrieval Using an Index *

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

Abstract

Region-based image retrieval systems aim to improve the effectiveness of content-based search by decomposing each image into a set of "homogeneous" regions. Thus, similarity between images is assessed by computing similarity between pairs of regions and then combining the results at the image level. In this paper we propose the first provably sound algorithm for performing region-based similarity search when regions are accessed through an index. Experimental results demonstrate the effectiveness of our approach, as also compared to alternative retrieval strategies.

1. Introduction

Many real world applications, in the field of medicine, weather prediction, and communications, to name a few, require efficient access to image databases based on content. To this end, the goal of content-based image retrieval (CBIR) systems is to define a set of properties (*features*) able to effectively characterize the content of images and then to use such features during retrieval. Users accessing a CBIR system often look for images containing particular "objects", possibly arranged in a specific spatial organization. To this end, in recent times, a number of *region-based* image retrieval systems has been presented [12, 4, 11, 1], which "fragment" each image into regions, i.e. sets of pixels sharing common visual characteristics, like color and texture. Conceptually, the process of similarity assessment between images can then be split into two distinct phases:

Matching Regions of the database image are associated to regions of the reference (query) image, by only considering "best" couplings (matches).

Combining The overall similarity between the query and a DB image is computed by combining similarity scores corresponding to matched regions (see Figure 1).

Region matching algorithms have only recently emerged as a need for CBIR systems. Existing systems, however, either use naïve heuristic matching algorithms when associating regions of the images being compared, thus obtaining incorrect results,[1] or consider a scenario, which is beyond the scope of our work, where spatial constraints are taken into account [3].

In this paper we will focus on k nearest neighbor queries, where the user asks for the k images in the database which are most similar to a query image. To speed-up query resolution, we present the first sound index-based algorithm for region-based image retrieval, and implement it into the WINDSURF system [1]. Our algorithm is independent of the underlying CBIR system, and only requires that similarity between images is computed from the similarity scores of matched regions. We begin our discussion by outlining limits in query processing of existing region-based image retrieval systems (Section 2). In Section 3 we precisely formulate the region matching problem, presenting an index-based approach ($\mathcal{A}_0^{WS}$) for its correct resolution. Section 4 presents experimental results, by looking at both efficiency and effectiveness issues, and Section 5 concludes.

2. Limits of Existing Systems

In order to give an adequate representation of obtained regions, existing region-based systems have focused only on the extraction of features, thus overlooking the important phase of query processing. As an example, in VisualSEEk [12] the similarity between two images is computed by taking into account color, location, dimension, and relative positioning of regions. Query processing, however, is carried out using simple heuristics: The matching phase is performed by issuing, for each region of the query image, a range query on color, location, and dimension requesting those regions whose similarity, with respect to the query region, is higher than a user-provided threshold; then, a *candidate set* of images is built, consisting in those images that present regions in all the result sets for the previous queries; finally, the optimum match is computed (combining phase)

* This work has been partially supported by InterData and ex-60% grants from MURST

[1] As an example, suppose that an user asks for an image containing two tigers: If a database image contains a single tiger, it is *not* correct to associate both query regions to the single "tiger" region of the DB image, since, in this case, information on the number of query regions is lost.

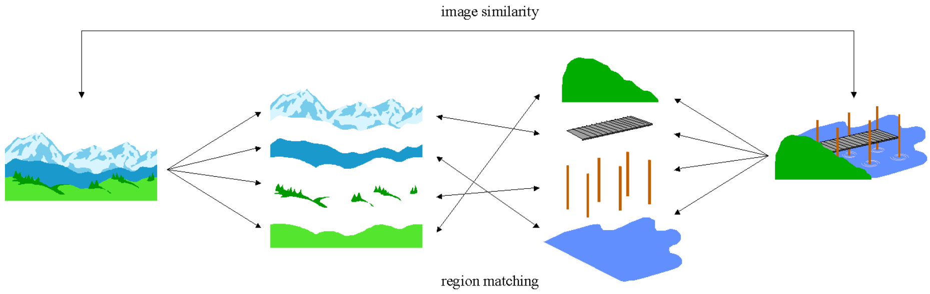


Figure 1. In region-based systems, similarity between images is assessed by taking into account similarity between matched regions.

on the set of candidate images. Thus, if a user would request for, say, the 10 images most similar to a given one, he/she is also forced to specify hard-to-guess similarity thresholds.

WALRUS (WAveLet-based Retrieval of User-specified Scenes) [11] is a region-based similarity retrieval system which fragments images using wavelets [7]. The matching phase of WALRUS consists in retrieving all the DB regions which are similar to at least one query region with a score $\geq \epsilon$. To this end, descriptors of DB regions are indexed using an R*-tree [2] and a range query, with a radius determined by ϵ, is issued for each query region. Then, in the combining phase, which is applied only to those images containing regions obtained in the matching phase, the relative sizes of matching regions are added up to obtain the overall similarity score between images. Images for which the similarity with the query image is higher than a user-specified ξ threshold are returned as the query result.

The main limitation of both VisualSEEk and WALRUS resides in the fact that they require the specification of similarity thresholds. Indeed, range queries are not well suited for the scenario we envision: 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 a similarity threshold; a high value for it could lead to an empty result, and slightly lowering this value could result in an overwhelming number of returned images.

Blobworld [4] is a CBIR system which fragments an image into regions (*blobs*), homogeneous with respect to color and texture, by using an Expectation-Maximization clustering algorithm. The Blobworld index-based query resolution algorithm uses an R-tree-like structure to index color descriptors of blobs. The matching phase is performed by requesting, for each blob in the query image, a predetermined number (in the order of the hundreds) of most similar DB regions by issuing a nearest neighbors query on the index. The combining phase only considers regions obtained in the matching phase and computes the overall image similarity using (weighted) fuzzy-logic operators to combine regions' scores. This approach has two major limitations. First, since best matches for query blobs are computed by ignoring matches for other blobs, a single blob in the database image can be associated to two distinct query blobs (see the "two tigers" example in Section 1). Second, the number of regions that are returned by the matching phase is a priori determined, thus it is unrelated to the number k of images requested by the user and to the specific query image. As we will show in Section 3, this can lead to miss the correct best images.

WINDSURF [1] is a region-based CBIR system that uses Discrete Wavelet Transform (DWT, [7]) and a fuzzy c-means algorithm to divide each image into regions. In WINDSURF, the similarity between two regions, R_{q_i} of a query image I_q and R_{s_j} of a database image I_s, is computed as:

$$r_{sim}(R_{q_i}, R_{s_j}) = h(d(R_{q_i}, R_{s_j})) \quad (1)$$

where $d()$ is a distance function, and $h()$ is a so-called *correspondence function* [6], mapping distance values to similarity scores, which satisfies the following properties: $h(0) = 1$ and $d_1 \leq d_2 \Rightarrow h(d_1) \geq h(d_2), \forall d_1, d_2 \in \Re_0^+$. In all the experiments, we used $h(d) = e^{-d/\sigma_d}$, where σ_d^2 is the variance of the distances computed over a sample of database regions. The overall distance between regions R_{q_i} and R_{s_j} takes into account both differences in color and texture descriptors and in their relative size (see [1] for details).

The combining phase consists in computing the overall similarity between images I_q and I_s as the average similarity between matched regions (with $\Gamma_s(R_{q_i})$ we denote the region R_{s_j} of I_s associated by the matching algorithm to the query region R_{q_i}):

$$I_{sim}(I_q, I_s) = \frac{1}{n} \sum_{i=1}^{n} r_{sim}(R_{q_i}, \Gamma_s(R_{q_i})) \quad (2)$$

If the match for a certain region R_{q_i} is undefined, i.e. the query region R_{q_i} is not associated to any region R_{s_j}, then it is $r_{sim}(R_{q_i}, \Gamma_s(R_{q_i})) = 0$.

3. Optimal Region Matching

In the following we consider that matching between regions is carried out by ignoring spatial constraints, i.e. only local

information (e.g. color and texture) about regions is taken into account when computing similarity between regions. We also assume that the overall similarity between images is computed by way of a monotonic function RM_{sim}. This is reasonable, since better matches between regions are expected to increase the overall similarity score. Note that both Blobworld and WINDSURF satisfy this requirement, since they use a combination of fuzzy-logic operations and an averaging operator, respectively.
The *optimal region matching* problem can then be formulated as a generalized assignment problem. Let $s_{ij} = r_{sim}(R_{q_i}, R_{s_j})$ be the similarity score between region R_{q_i} of $I_q = \{R_{q_1}, \ldots, R_{q_n}\}$ and region R_{s_j} of $I_s = \{R_{s_1}, \ldots, R_{s_m}\}$, and denote with $\mathcal{H}$ an index set of matched regions, $\mathcal{H} = \left\{(i,j)|R_{s_j} = \Gamma_s(R_{q_i})\right\}$; of course, it is $|\mathcal{H}| \leq \min\{m,n\}$. The goal is to maximize the function $RM_{sim}(s_{i_1j_1}, \ldots, s_{i_{|\mathcal{H}|}j_{|\mathcal{H}|}})$, with $(i_hj_h),(i_lj_l) \in \mathcal{H}, (i_hj_h) \neq (i_lj_l)$. To this end, we introduce the variables x_{ij}, where $x_{ij} = 1$ if $R_{s_j} = \Gamma_s(R_{q_i})$, $x_{ij} = 0$ otherwise.

$$I_{sim}(I_q, I_s) = \max\ RM_{sim}(s_{i_1j_1}, \ldots, s_{i_{|\mathcal{H}|}j_{|\mathcal{H}|}}), \quad (i_hj_h),(i_lj_l) \in \mathcal{H}, (i_hj_h) \neq (i_lj_l) \tag{3}$$

$$\mathcal{H} = \{(i,j)|x_{ij} = 1\} \tag{4}$$

$$\sum_{j=1}^{m} x_{ij} \leq 1 \ (i = 1, \ldots, n), \tag{5}$$

$$\sum_{i=1}^{n} x_{ij} \leq 1 \ (j = 1, \ldots, m), \tag{6}$$

$$x_{ij} \in \{0,1\} \ (i = 1, \ldots, n)(j = 1, \ldots, m) \tag{7}$$

Eq. 3 means that to determine the overall score $I_{sim}(I_q, I_s)$ we have to consider only the matches $\Gamma_s()$ in $\mathcal{H}$ (Eq. 4). Eq. 5 (Eq. 6) 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).

Definition 3.1 (Correct matching) *A set of x_{ij} values that satisfies the constraints expressed by Eqs. 5, 6, and 7 is called a* correct *matching.*

Definition 3.2 (Complete matching) *A correct matching for which it is $\sum_{j=1}^{m} x_{ij} = 1, (i = 1, \ldots, n)$ (i.e. each query region is associated to a region of the database image) is called a* complete *matching.*

It should be noted that *any* correct matching for a database image having a number of regions lower than that of the query regions is obviously not complete.

Definition 3.3 (Optimal matching) *The correct matching that maximizes the function expressed by Eq. 3 is called the* optimal *(or* exact*) matching, and will be denoted as $\Gamma_s^{opt}()$.*

3.1. Index Evaluation

In this Section we describe an index-based algorithm aiming to speed-up the evaluation of k nearest neighbor queries. This is carried out by reducing the number of *candidate images*, i.e. images on which the optimal region matching problem has to be solved.
Since similarity between images is computed by combining distances between regions' features, our approach uses a distance-based access method (DBAM), like R*-tree [2] or M-tree [5], to index regions contained in database images. Such index structures are able to efficiently answer k nearest neighbor queries, as well as to perform a *sorted access* to the data, i.e. to output regions one by one in increasing order of distance with respect to a query region [9].
To retrieve best matches for query regions, we run a sorted access to the indexed regions for each region in the query image. The problem, here, is to specify a suitable condition to stop such sorted accesses when we are guaranteed that the best k images could be retrieved by only taking into account regions returned by the index. Then, once the sorted access phase has been stopped, the optimal region matching problem is solved for each *candidate image*, i.e. for each image having at least one region which has been retrieved during the sorted access phase, and the best k images are returned as the result (Figure 2).

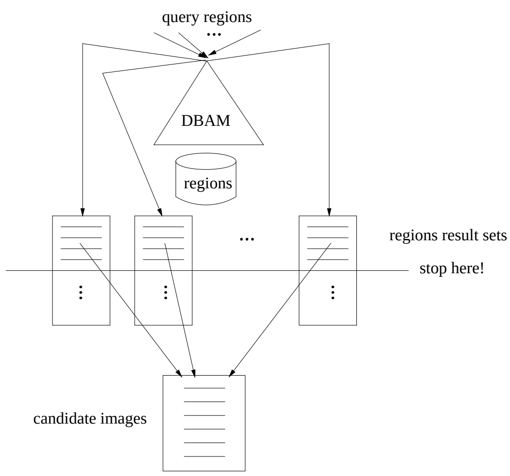


Figure 2. Producing the candidate set of images from the sorted access phase.

To ensure that the best k results are included into the set of candidate images, 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. To this end, the stopping condition should take into account the correctness of regions' assignments (see Definition 3.1), i.e. the sorted access phase is halted when optimal matches for non-candidate images

could only lead to lower scores with respect to the k-th best correct match for candidate images, computed by only taking into account regions retrieved by the index.
Consider, as an example, the case where $n = 2$ and $k = 1$, with similarity scores obtained by sorted accesses to a DBAM given in Table 1, and suppose that we compute image to image similarity by taking the average of regions' similarity. After the first step, the candidate set of images is $\{I_1, I_3\}$ with overall scores $\frac{0.9+0}{2} = 0.45$ and $\frac{0+0.87}{2} = 0.435$, respectively. Since an image outside the candidate set could potentially lead to an overall score of $\frac{0.9+0.87}{2} = 0.885$, 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 $\frac{0.85+0}{2} = 0.425$ (remember that region R_{2_1} can match at most one region of I_q); therefore, the sorted accesses cannot yet be stopped. At the third step, also image I_4 is added to the candidate set, having a score of $\frac{0.83+0}{2} = 0.415$. Finally, at fourth step, we obtain a complete matching (see Definition 3.2) 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 $\frac{0.71+0.87}{2} = 0.79$. 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 $\frac{0.71+0.72}{2} = 0.715$). The monotonicity of the combining function RM_{sim} is used here to ensure algorithm correctness. 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 region matching problem on the set of candidate images, we need to compute similarity scores between query regions and *all* the regions of candidate images.

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 1. A sorted access example for a query image with two regions: R_{q_1} and R_{q_2}.

From above example, it is clear that the sorted access phase can be stopped as soon as a complete assignment is found, taking into account only regions returned by index scans.[2] This leads to the $\mathcal{A}_0^{WS}$ algorithm shown in Figure 3.
The *random access* phase consists in computing those similarity scores s_{ij} between query regions and regions of candidate images not returned in the X^i regions result sets. Finally, to compute an optimal assignment for a candidate image I_s, in WINDSURF we can use the Hungarian algorithm [10], since the generalized assignment problem reduces to the linear Assignment Problem due to Eq. 2.

```
A_0^WS(I_q: query image, k: integer, T: DBAM)
{ ∀ region R_qi 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 assignment exists,
    considering only regions in ∪_i X^i;
  ∀ image I_s having regions in ∪_i X^i,
    ∀ pair R_qi, R_sj
      if R_sj ∉ 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 3. The $\mathcal{A}_0^{WS}$ algorithm.

Correctness of $\mathcal{A}_0^{WS}$ (proof is omitted for lack of space) is independent of the specific RM_{sim} function used to combine regions' scores into similarity between images, since it only relies on the monotonicity of RM_{sim}.
It can be noted that sorted and random access phases of $\mathcal{A}_0^{WS}$ somewhat resemble those of Fagin's $\mathcal{A}_0$ algorithm [8], the major difference being that $\mathcal{A}_0$ does not deal with the issue of correct matching. As an example, in Table 1, $\mathcal{A}_0$ would incorrectly stop sorted access after step 2.

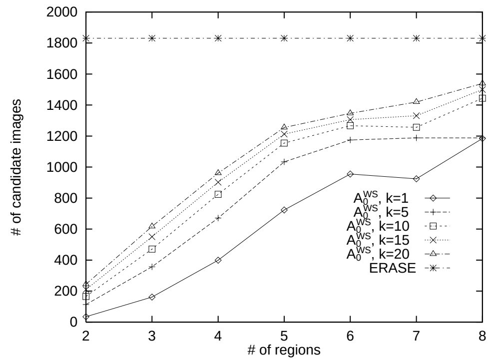
4. Experimental results

Preliminary experimentation of proposed techniques has been performed on the WINDSURF system, using a sample medium-size data-set consisting of about 2000 real-life images from the *IMSI-PHOTOS* CD-ROM.[3] The over 8000 obtained regions were indexed using an M-tree [5]. The query workload consists in about one hundred randomly chosen images not included in the data-set. All experiments were performed on a Pentium II 450 MHz PC with 64MB of main memory running Windows NT 4.0.
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 4 (a) we compare the average 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, for different values of k. Of course, a sequential solution for the query would lead to a number of candidate images equal to the number of images in the data-set (the horizontal line labeled ERASE, for Exact Region Assignment SEquential algorithm), whereas for the index version this number depends both on k and on of the number of query regions. As the graph shows, the $\mathcal{A}_0^{WS}$ algorithm does well in reducing the number of candidate images. Clearly, its performance degrades as the number of query regions increases, since

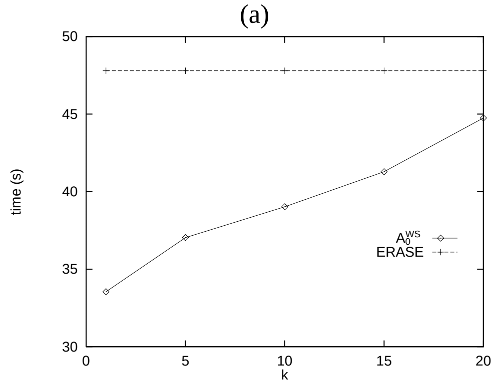
[2] By the way, this is the reason why Blobworld algorithm is not correct, since its stopping condition cannot guarantee the existence of a complete assignment.

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

the complexity of finding k objects in the intersection of n sets augments with n. This is also confirmed by Figure 4 (b), where query response times are shown.



(a)



(b)

Figure 4. Average number of candidate images vs. number of query regions (a), and response time vs. k ($n = 3$) (b).

To show the effectiveness of our approach, we compare results obtained by the $\mathcal{A}_0^{WS}$ algorithm when only a fraction of query regions is used to query the database which leads to *approximate* queries. In Figure 5 the tradeoff between quality of the result and query evaluation cost is shown. Quality is measured as the sum of similarity scores for the k best images normalized with respect to the case where all regions of the query are used. Cost is computed as the elapsed time relative to the time needed for resolving the "all regions" query. The graph clearly shows that quality and cost are strictly correlated in that both decrease when the number of query regions reduces. As a further observation, since the major part of the points falls below the "relative cost=quality" line, an effective way to reduce query costs is to use only some of the regions in the query image.

5. Conclusions

In this work we have introduced an original approach to correct resolution of similarity queries for region-based image

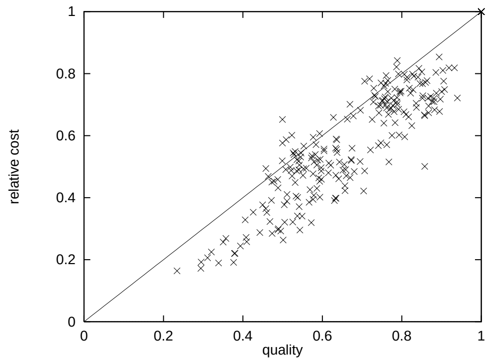


Figure 5. Tradeoff between quality and cost for approximate queries ($k = 15$).

retrieval. In particular, an index-based algorithm ($\mathcal{A}_0^{WS}$) has been presented which computes the optimal matching between regions of the query image and regions of a database image, in order to maximize the overall similarity score between images. Preliminary experiments conducted over the WINDSURF system have shown that our approach is indeed very effective with respect to alternative retrieval strategies.

References

[1] S. Ardizzoni, I. Bartolini, and M. Patella. Windsurf: Region-based image retrieval using wavelets. In *IWOSS'99*, pages 167–173, Florence, Italy, Sept. 1999.

[2] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. The R*-tree: An efficient and robust access method for points and rectangles. In *SIGMOD'90*, pages 322–331, Atlantic City, NJ, May 1990.

[3] S. Berretti, A. D. Bimbo, and E. Vicario. Managing the complexity of match in retrieval by spatial arrangement. In *ICIAP'99*, Venezia, Italy, Sept. 1999.

[4] C. Carson, M. Thomas, S. Belongie, J. M. Hellerstein, and J. Malik. Blobworld: A system for region-based image indexing and retrieval. In *VISUAL'99*, pages 509–516, Amsterdam, The Netherlands, June 1999.

[5] P. Ciaccia, M. Patella, and P. Zezula. M-tree: An efficient access method for similarity search in metric spaces. In *VLDB'97*, pages 426–435, Athens, Greece, Aug. 1997.

[6] P. Ciaccia, M. Patella, and P. Zezula. Processing complex similarity queries with distance-based access methods. In *EDBT'98*, pages 9–23, Valencia, Spain, Mar. 1998.

[7] I. Daubechies. *Ten Lectures on Wavelets*. SIAM, 1992.

[8] R. Fagin. Combining fuzzy information from multiple systems. In *PODS'96*, pages 216–226, Montreal, Canada, June 1996.

[9] G. R. Hjaltason and H. Samet. Ranking in spatial databases. In *SSD'95*, pages 83–95, Portland, ME, Aug. 1995.

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

[11] A. Natsev, R. Rastogi, and K. Shim. WALRUS: A similarity retrieval algorithm for image databases. In *SIGMOD'99*, Philadelphia, PA, June 1999.

[12] J. R. Smith and S.-F. Chang. VisualSEEk: A fully automated content-based image query system. In *ACM Multimedia '96*, pages 87–98, Boston, MA, Nov. 1996.