Image Retrieval by Examples

Roberto Brunelli and Ornella Mich

***Abstract*—A currently relevant research field in information sciences is the management of nontraditional distributed multimedia databases. Two related key issues are to achieve an efficient content-based *query by example* retrieval and a fast response time. This paper presents the architecture of a distributed image retrieval system which provides novel solutions to these key issues. In particular, a way to quantify the effectiveness of low level visual descriptors in database query tasks is presented. The results are also used to improve the system response time, which is an important issue when querying very large databases. A new mechanism to adapt system query strategies to user behavior is also introduced in order to improve the effectiveness of relevance feedback and overall system response time. Finally, the issue of browsing multiple distributed databases is considered and a solution is proposed using multidimensional scaling techniques.**

***Index Terms*—Clustering, distributed database, image database browsing, image retrieval, multidimensional scaling, relevance feedback.**

I. Introduction

THE CURRENT ever growing amount of multimedia data requires a large integrated effort in the research fields of computer vision, information retrieval and database management for its effective management. In particular, retrieving information from multimedia repositories requires the development of techniques to supplement traditional methods based on textual descriptions and searches. The reason for this necessity is twofold: associating textual descriptions to multimedia data can be very expensive, and, more importantly, textual descriptions may not characterize data adequately for subsequent retrieval.

An attempt to overcome this limitation is through *query by example* where non textual queries are formulated by the user using multimedia items related to the material he/she is looking for, e.g., images or video clips for searching footage [1]–[6].

In the *query by example* framework, the user formulates a query by providing examples of objects similar to the one he/she wishes to retrieve. The system converts them into an internal representation used for assessing their similarity to the items stored in the database to be searched. The main advantage of *query by example* is that the user is not required to provide an explicit description of the items, which is instead computed by the system. In order for this paradigm to be effective, good content descriptions must be computed automatically by the system and ways to compare them obtaining results in accordance with human judgment should also be available.

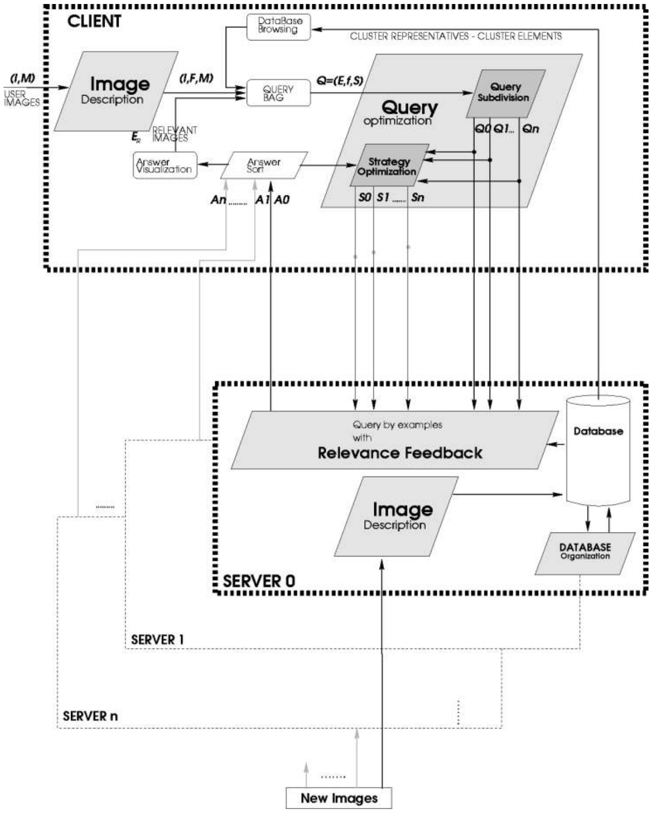

Fig. 1. General architecture for an image retrieval system based on the query by example paradigm. The shaded blocks are considered in detail by the current paper.

This paper discusses the use of pattern analysis techniques, such as density estimation, clustering, and multidimensional scaling, for the development of a computer assisted image search system, COMPASS.

The architecture of the system is described in Section II. The issue of small yet effective image descriptors is considered in Section III, while different query strategies with relevance feedback are described in Section IV. Algorithms for optimizing search strategies are presented in Section V. Finally, some browsing issues are considered in Section VI and the concluding remarks are reported in Section VII.

II. System Architecture

The overall structure of COMPASS, an image retrieval system to support the query by example paradigm for multiple distributed databases, is presented in Fig. 1. The system is configured as a client–server architecture in which a client

Manuscript received October 26, 1999; revised May 10, 2000. This work was supported in part by CEC under Esprit Project 25389 OPAL. The associate editor coordinating the review of this paper and approving it for publication was Dr. Anna Hac.

The authors are with the Interactive Sensory Systems Division, ITC-irst, Povo 38050, Italy (e-mail: brunelli@itc.it, mich@itc.it).

Publisher Item Identifier S 1520-9210(00)07225-4.

application can submit a user query to multiple image servers. The answers from multiple image servers are then merged and proposed to the user as a single result.

Following the query by example paradigm, users rely on the images themselves to formulate queries. A generic image $\mathcal{I}$ is characterized as a triple $(\boldsymbol{I}, \boldsymbol{F}, \boldsymbol{M})$ whose elements represent a complete description of the image pixels $\boldsymbol{I}$, possibly indirectly by pointing to the corresponding memory storage, a derived feature description $\boldsymbol{F} = \{\boldsymbol{F}_i\}$, automatically computed by the system, and associated meta data $\boldsymbol{M}$ providing information on image contents. Derived image descriptions can be computed directly by the client application while meta information is not usually provided automatically.

A query by example $\boldsymbol{Q}$ is defined by giving a set $\boldsymbol{E}$ of images and, possibly, by selecting a subset $\boldsymbol{f}$ of $\boldsymbol{F}$ and a comparison strategy $\boldsymbol{S}$ to be used by the image servers when comparing the query images to those stored in the database:

$$\boldsymbol{Q} = (\boldsymbol{E}, \boldsymbol{f}, \boldsymbol{S}). \tag{1}$$

In order to answer a query, the image server compares the images in the query set $\boldsymbol{E}$ to the stored ones using strategy $\boldsymbol{S}$, obtaining a *dissimilarity* score for each of them. The dissimilarity of images could be computed using both the derived descriptors $\boldsymbol{f}$ and image meta-data $\boldsymbol{M}$. Derived descriptors are often represented as numerical vectors while meta data is usually in textual form. The analysis presented in this paper will be limited to the use of derived descriptors represented as numerical vectors, leaving out any available meta data in the computation of image similarity. As the set of query images must be compared to other images, a function to compute the dissimilarity of an image from an image set must be introduced. If we restrict ourselves to metric spaces for the derived descriptors, the distance between an image I and an image set $\boldsymbol{E}$ can be computed using the following formula:

$$D(\mathcal{I}, \boldsymbol{E}) = \min_{\mathcal{I}' \in \boldsymbol{E}} d(\mathcal{I}, \mathcal{I}') \tag{2}$$

where d represent the distance defined in the metric space.

The effectiveness of feature comparison is improved by the use of *relevance feedback* which modifies the distance of the metric space using information derived from the interaction of the user with the system. The servers then sort database items by increasing dissimilarity. The set $\boldsymbol{A}$ of the top ranked ones is returned to the client together with their dissimilarity value and, if available and requested, associated meta-data. The client, upon receiving the answer from each server, sorts the resulting complete set by dissimilarity and offers to the user a single answer.

The interaction of the user with the client is based on a graphical interface (see Fig. 2), which, in close resemblance to the interfaces for querying traditional databases, provides: an area for the specification of the query; the possibility of restricting searchable image content, i.e., image descriptors; an area where retrieved items are displayed; a way to limit the number of items retrieved by imposing a threshold on the minimum required image similarity.

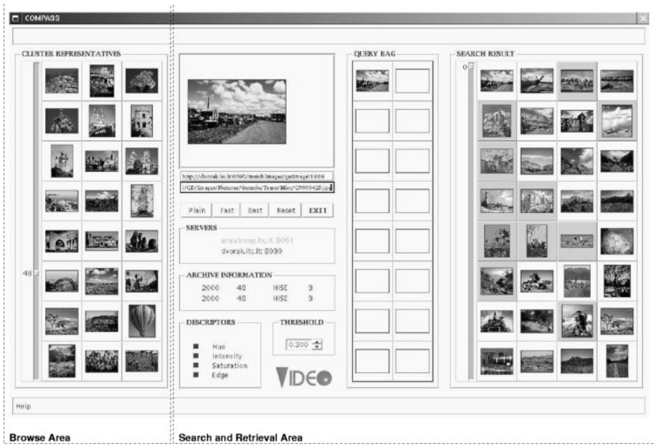

Fig. 2. COMPASS client GUI. The areas corresponding to the different functionalities of the client are outlined.

Besides database query by example, COMPASS also supports another very important activity: database browsing. This operation is important in the case of multimedia databases used as repositories of material to be creatively (re)assembled into new multimedia products such as cd-rom, composite images, footage, etc. Browsing support requirements differ from those of querying: database items should be organized and presented to the user in such a way that exploration of content is possible. The solution investigated in COMPASS is the organization of databases in clusters of similar images (see Section VI). Each cluster is represented by the client with a *key image* and cluster elements can be displayed on user demand providing a more detailed view of available images.

III. Image Description

One of the key issues in querying image databases by similarity is the choice of appropriate image descriptors and corresponding similarity measures. In a recent paper [7] the problem of quantifying the effectiveness of several low level visual descriptors was addressed. The proposed solution relies on the following definition:

Definition 1: Given an n-dimensional histogram space $\mathcal{H}$ and a dissimilarity measure[1] d on $\mathcal{H}$, the *capacity curve* C of $\mathcal{H}$ is defined as the density distribution of the dissimilarity between the two elements of all possible histogram couples within $\mathcal{H}$.

Histogram capacity curves provide a basis on which the effectiveness, i.e., the discrimination ability, of different image descriptors can be compared. The shape of $C(t)$ is an indicator of the distribution of histograms in $\mathcal{H}$ with the topology induced by the selected comparison dissimilarity measure. If the average value of dissimilarity is low, histograms are not sparse enough in $\mathcal{H}$ and histogram indexing is not effective. This can be formalized by the following definition:

Definition 2: The *indexing effectiveness* $\mathcal{E}$ of an histogram space $\mathcal{H}$ is given by the average dissimilarity value:

$$\mathcal{E} = \int yC(y)\, dy. \tag{3}$$

The indexing effectiveness $\mathcal{E}$ can be used to assess several descriptor-dissimilarity combinations for image retrieval

[1] In this context, a dissimilarity measure is a bounded, positive, and symmetric function defined over a subset of $\boldsymbol{R}^n \times \boldsymbol{R}^n$.

TABLE I
THE EFFECTIVENESS $\mathcal{E}$ OF SOME LOW-LEVEL VISUAL DESCRIPTORS. EDGENESS IS DEFINED AS THE MAGNITUDE OF THE LUMINANCE GRADIENT WHILE THE CO-OCCURENCE OF HUE OR LUMINANCE IS A TWO DIMENSIONAL HISTOGRAM OBTAINED BY PARTITIONING THE IMAGE SPACE INTO COUPLES OF PIXELS BY MEANS OF A BINARY SPATIAL RELATION: THE DESCRIPTOR VALUES AT THE TWO PIXELS ARE USED AS INDICES IN A 2-D HISTOGRAM

Descriptor	$\mathcal{E}_{\mathtt{VIDEO}}$	$\mathcal{E}_{\mathtt{STILLS}}$
Co-occurrence (hue)	57	68
Hue	55	70
Co-occurrence (lum)	52	50
Luminance	43	46
Edgeness	22	32

applications. Several ways to compute image dissimilarity were considered in [7]: χ^2, Kolmogorov–Smirnov, Kuiper, and L_p norms. The L_1 norm provided the best overall results in terms of indexing effectiveness and stability with respect to the number of histogram bins used. The main findings of [7] on the discrimination ability of some basic image descriptors on two different databases (VIDEO, a set of 40 000 frames from nine different video clips, and STILLS, a set of 3500 still images from a commercial collection) are summarized in Table I.

The effectiveness of the different descriptors can also be used to optimize the order in which they are compared. In image retrieval tasks, a threshold on the minimum acceptable similarity is usually imposed to limit the number of retrieved items. The computation of image dissimilarity can be stopped as soon as its monotonically increasing value exceeds the retrieval threshold. When multiple histograms are used to characterize an image, they can be concatenated in many different ways to obtain a single numerical vector describing the image. However, the order in which histograms are concatenated impacts on the performance of the system. Comparing the descriptors sorted by decreasing effectiveness is expected to increase the computational savings associated with the use of a retrieval threshold. Experiments on the same data used in [7] are reported in Fig. 3 and confirm this expectation.

IV. QUERY BY EXAMPLES WITH RELEVANCE FEEDBACK

Relevance feedback is a fundamental mechanism by which system response can be improved by using information fed by the user [8]–[11]. Whenever the system presents to the user a set of images considered to be similar to the provided examples, the user can pick among them the most *relevant* to the submitted query and add them to the original query. The resulting extended set $\boldsymbol{E}_R$ can be used to improve system response in a variety of ways [11]. A common approach to the implementation of relevance feedback for a system using image descriptors in numerical form is that of feature weighting and is based on the vector model used for textual documents.

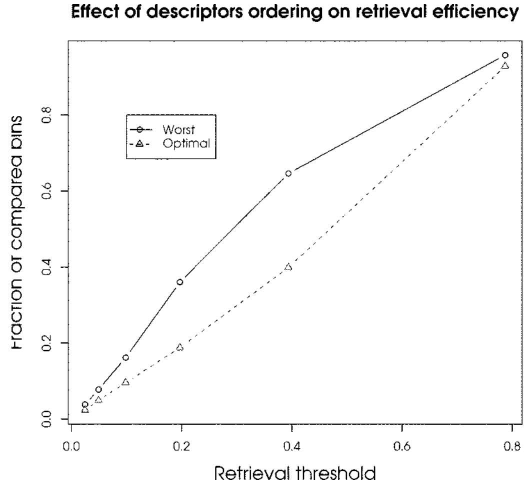

Fig. 3. Expected gain in speed resulting from properly sequencing the image descriptors before comparing them. Note the significant advantage over the worst case, where the order in which the descriptors are used is inversely proportional to their capacity.

As COMPASS image derived descriptors $\boldsymbol{F} = \{\boldsymbol{F}_i\}$ are histograms with the same number of bins and normalization, the dissimilarity of two images can be computed by

$$d(\mathcal{I}, \mathcal{I}') = \sum_i \sum_j w_{ij} |F_{ij} - F'_{ij}| \tag{4}$$

where i represents the ith descriptor and j the value of the jth bin of the descriptor. This distance introduces a metric structure in the derived descriptors space and can be used to compute the distance of the query set $\boldsymbol{E}_R$ from each database item using the formula reported in (2).

The set $\{w_{ij}\}$ is computed by the system and is used to incorporate relevance feedback into the comparison metric. Relevant images should be similar to each other for some of the components of their descriptors F_{ij}. This means that the standard deviations σ_{ij} computed over set $\boldsymbol{E}_R$ should be small for the components capturing the similarity of the images and larger for the components which are *not relevant.* In this paper the following family of weighting schemes is introduced to emphasize distances along the appropriate directions

$$w_{ij} = k_\beta \frac{1}{\sigma_{ij}^\beta} \tag{5}$$

where $k_\beta = \left(\sum_{lm} \sigma_{lm}^{-\beta}\right)^{-1}$ is a normalizing factor, and β is a parameter which modulates the weighting effect. These weighting schemes are a generalization of the single scheme proposed in [12]. There are some major drawbacks to the use of (5):

- the use of σ_{ij} tacitly assumes that the images in the query represent a compact set with ellipsoidal shape;
- the comparison metric is modified in the same way over the descriptor space;
- the time necessary for the computation of D depends on the number of images in the query set $\boldsymbol{E}$.

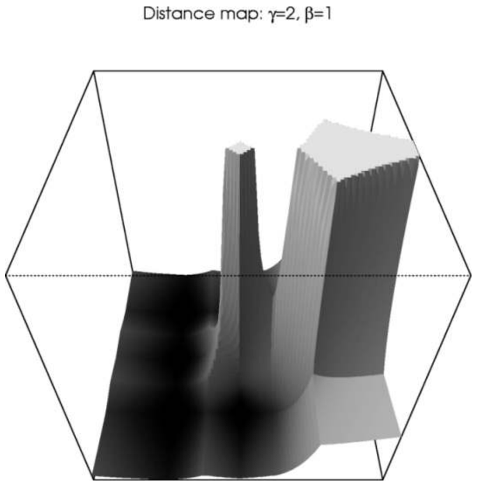

Fig. 4. Effect of the introduction of negative examples on the computation of distances. Negative examples define repulsive regions in pattern space by modifying the metric used in the computation of image distances. The size of the repulsive regions is controlled by the value of γ.

Furthermore, the amount of weighting specified by β is expected to be query dependent and should be optimized on a case-by-case basis. A way to overcome these drawbacks is presented in Section V.

An interesting addition to relevance feedback for image retrieval comes from the introduction of *negative examples.* An approach based on the generalization of Boolean searches has been presented in [13] relating image distances to fuzzy predicates. In this paper a new approach is introduced using negative examples as a perturbation in the metric used to compute the image distances. The user, by providing positive examples, implicitly defines a generalized Boolean query whose value is given by normalized image distances. However, when the images looked for are organized in complex arrangement in the descriptor space, computing image similarity with (4) and (5) may result in persistent *irrelevant* images, i.e., *negative examples.* Knowledge of which of the retrieved images are not relevant to the current query, can be used to better characterize the regions of the descriptor space which contain relevant images by creating *negative* regions carving complex geometries in feature space. The set $\boldsymbol{E}_{\hat{R}}$ of irrelevant images retrieved by the system can be used to introduce a modified dissimilarity function D'

$$D'(\mathcal{I}, \boldsymbol{E}_R, \boldsymbol{E}_{\hat{R}}) = D(\mathcal{I}, \boldsymbol{E}_R)\left[\frac{D(\mathcal{I}, \boldsymbol{E}_R)}{D(\mathcal{I}, \boldsymbol{E}_{\hat{R}})}\right]^{\gamma} \qquad (6)$$

where γ represents the intensity of the action of $\boldsymbol{E}_{\hat{R}}$ and $\boldsymbol{E}_R$ represents the set of relevant images (i.e., the *positive examples*). As set $\boldsymbol{E}_{\hat{R}}$ is close to $\boldsymbol{E}_R$, for points lying far from $\boldsymbol{E}_{\hat{R}}$ and $\boldsymbol{E}_R$ in feature space $D \approx D'$, for points nearer to $\boldsymbol{E}_R$ than to $\boldsymbol{E}_{\hat{R}}$ the original dissimilarities are reduced, and for points nearer to $\boldsymbol{E}_{\hat{R}}$ than $\boldsymbol{E}_R$ they are increased. A visual presentation of this effect is shown in Fig. 4.

V. Query Optimization

The drawbacks associated with the use of (5) on the effectiveness and efficiency of relevance feedback can be minimized by

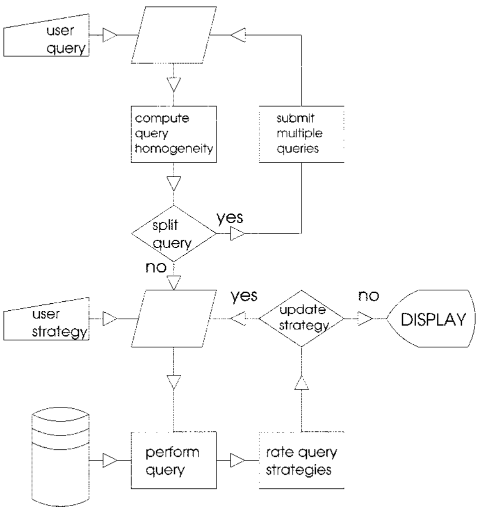

Fig. 5. Flow chart of the proposed query supervisor agent.

- determining whether the specified query set $\boldsymbol{Q}$ while not being compact itself, is composed of two or more compact sets: the query could then be split into simpler subqueries, each of them better suited to the use of (5).
- condensing the query set using a smaller number of images while preserving the effectiveness of the original set;
- adaptively choosing β, the parameter that modulates the amount of metric change.

A block structure of the resulting query optimization module is reported in Fig. 5, while the following sections introduce the necessary pattern analysis techniques.

A. Query Subdivision

The cloud of points representing the query images in the descriptor space may exhibit local grouping, i.e., clusters, suggesting the splitting of the original query set into multiple subsets, each of them characterized by the images belonging to one of the clusters.

From a data analysis perspective, the relevant issue is whether the structure of the point distribution supports the presence of multiple clusters or not. There are no completely satisfactory methods to determine the number of clusters for any type of cluster analysis [14], [15]. The situation analyzed by the current paper presents additional difficulties due to the small number of images used to define the query: no asymptotic results can be used, and methods relying on density estimates can not be applied. The proposed strategy is based on two steps:

1) establish whether the original query should be split or not;
2) if the original query should be split determine the number of clusters into which it should be split.

The first step is based on the use of a statistic originally proposed by Duda and Hart [16]. Let us denote with $d(\mathcal{I}, \mathcal{I}')$ the distance between the descriptors of two images $\mathcal{I}$, $\mathcal{I}'$ and with $J(c)$ the clustering criterion function for c clusters $C_1, \cdots, C_c$:

$$J(c) = \sum_{i=1}^{c} \sum_{\mathcal{I} \in C_i} d(\mathcal{I}, \boldsymbol{m}_i) \qquad (7)$$

where $\boldsymbol{m}_i$ is the *central* image of the ith cluster. The quantity $J(c)$ is a random variable whose average value decreases monotonically with c. In particular, if data are organized into $\hat{c}$ compact, well separated clusters, the value of $J(c)$ is expected to decrease rapidly until $\hat{c} = c$, and much more slowly thereafter. Knowledge of the distribution of $J(2)/J(1)$ under the null hypothesis that all samples belong to a single cluster forms the basis for a test to reject or accept the null hypothesis. Unfortunately, analytical results are often not available. An approximate result is derived in [16] when the distance used in the comparison is the Euclidean norm. As the comparison metric used by COMPASS is the L_1 norm for which results are much harder to obtain, a Monte Carlo approach was chosen [17]. As detailed in Section III, each image is represented by histograms of several low level visual features, normalized to unit. In order to determine the distribution of $\mathcal{J} = J(2)/J(1)$ for different sample sizes n (from six to 16), 10 000 random samples were generated, satisfying the image descriptors constraints: number of features, number of bins, and normalization to unit. For each random sample the Linde–Buzo–Gray clustering algorithm [18] using the L_1 metric was applied ten times to find the optimal two cluster partition. The corresponding values of $\mathcal{J}$ were then used to compute the required distributions. The empirical distributions for different values of n are markedly different, n being too small to ensure an asymptotic regime.

Given a set of N query images the value of $\mathcal{J}$ is computed: if the null hypothesis of a single cluster can be rejected with the prescribed confidence, the appropriate number of clusters should then be determined. The most appropriate number of subqueries into which the original query should be split is determined by the so called *silhouette* coefficient $\mathcal{S}$ introduced in [19]. Let us introduce the following quantities:

$$a(i) = \frac{1}{n_{C(i)}} \sum_{j \in C(i),\, j \neq i} d(\mathcal{I}_i, \mathcal{I}_j)$$

$$b(i) = \min_{C \neq C(i)} \frac{1}{n_C} \sum_{j \in C} d(\mathcal{I}_i, \mathcal{I}_j)$$

where n_C is the number of elements in cluster C; the *silhouette* of element i is then defined as

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}. \tag{8}$$

When a cluster contains a single object, $s(i) = 0$. The higher the value of $s(i)$ the stronger the *membership* of i to its corresponding cluster. Elements that can not be clearly assigned to any cluster have a silhouette value near to zero. The silhouette coefficient $\mathcal{S}$ is then defined as

$$\mathcal{S} = \frac{1}{N} \sum_{i=1}^{N} s_i. \tag{9}$$

The value of $\mathcal{S}$ is bound to the closed interval $[-1, 1]$: the higher the value the better the overall classification of data for the given clustering. Furthermore, $\mathcal{S}$ is a dimensionless quantity that does not change when the distances between samples are multiplied by a constant factor. The knowledge of the silhouette coefficient can be used to choose an appropriate number of clusters $\hat{k}$ such that

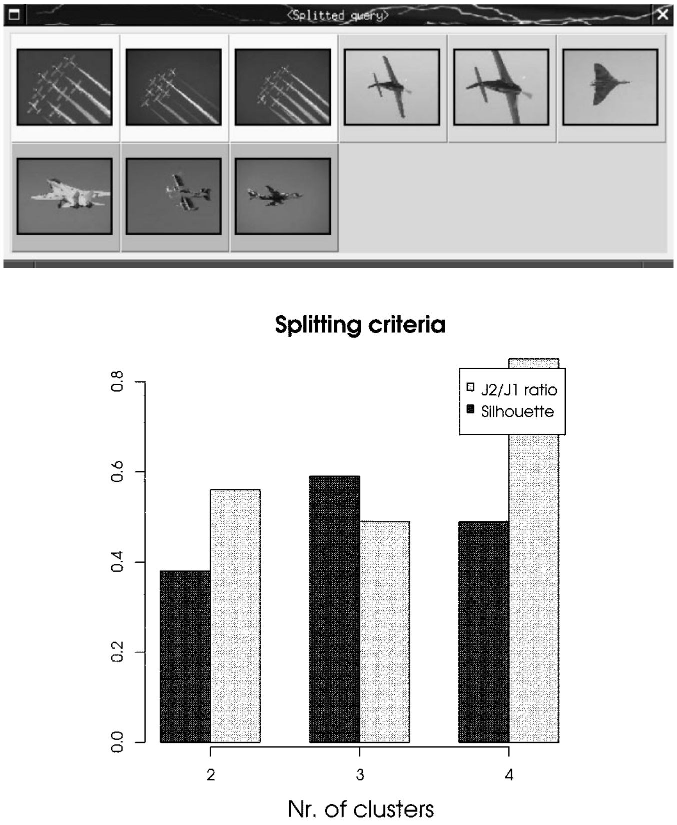

Fig. 6. Sample query and the way it would be split by the system.

$$\mathcal{S}(\hat{k}) = \max_{k=2, \cdots, K} \mathcal{S}(k). \tag{10}$$

The above computations are used to subdivide the original query images into several, simpler queries, each of which is better conditioned for the application of relevance feedback mechanisms (see Fig. 6).

The resulting simplified queries are then submitted to the image databases. For each simplified query a new comparison metric is computed according to (5). As a result, the metric used for image comparison is no longer a uniform modification of the unweighted distance: each subquery locally modifies the comparison metric, overcoming one of the limitations of the original feature weighting approach.

B. Strategy Optimization

Splitting the original query into smaller ones does not impact directly on the complexity of the computation of D and does not provide any hint on the optimal value of β. However, the system, upon receiving user feedback, can automatically compare different query strategies by looking at the ranking of the images selected by the user in the corresponding answers: the lower the average rank, the better the strategy. Note that this is quite different from the approach introduced in [20] where the user is required to rank all the images returned by the system.

Two aspects characterize the choice of the optimal search strategy: the determination of the best query representation and the selection of the optimal β value. In the following analysis only two representations for the (positive) query set are considered: the original set $\boldsymbol{E}_R$ and a condensed set obtained replacing the images in $\boldsymbol{E}_R$ with a *virtual* image represented by the arithmetic average of the descriptors in set.

For each representation of the query set, different values of β provide different strategies: the resulting set of dissimilarity functions constitutes the *optimization* space from which the best comparison method must be chosen.

Replacing the original query set with a single, *virtual* average image reduces the amount of computation required to estimate the distance of each database image from the query images. However, this simplification may not be always appropriate. As an example, if the original query set is not convex, the results may be meaningless, as the average image could be located in a region of feature space which is not representative of the original set. While query subdivision reduces these problems, the possibility of using the condensed representation should be assessed more directly. Since the purpose of using such a representation is to speed up the computation while obtaining, essentially, the same results associated to the original, complete set, it is necessary to verify that the two representations yield very correlated answers.

At each interaction, the system returns an image set $\boldsymbol{A}$ with the N database images most similar to the submitted query. Using this restricted number of images, it is possible to decide which representation of the query set is most efficient for the given query. This can be done by simulating system response using the restricted set $\boldsymbol{A}$ as image database. If the responses using the full and condensed representations are strongly correlated, the condensed representation is to be preferred being faster. Let us see how this correlation can be computed.

Each image in $\boldsymbol{A}$ can be characterized with a couple of values (D_i^c, D_i^f) representing its dissimilarity to the query using the condensed and full representations respectively. The set $\{(D_i^c, D_i^f)\}_{i=1,\cdots,N}$ can be considered as a random sample from a population with a bivariate distribution function. Let A_i be the rank of D_i^f among $D_1^f, \cdots, D_N^f$ when they are arranged in descending order, and B_i the rank of D_i^c among $D_1^c, \cdots, D_N^c$ defined similarly to A_i. The amount of correlation of the two answers can then be quantified by their Spearman rank correlation coefficient [21]. An important characteristic of rank correlation is its nonparametric nature. To assess the significance of a correlation value it is not necessary to know the bivariate distribution from which (D_i^c, D_i^f) are drawn. Given a confidence level, it is possible to decide whether the complete and condensed query representations results are sufficiently correlated or not.

The next step in choosing the optimal strategy is the selection of β. In order to adapt its value to the query more information is needed. The required additional data are provided by the user him(her)self with the selection of relevant images from set $\boldsymbol{A}$. Let us restrict to a discrete set $\{\beta_j\}$ of possible values. For each value β_j a query can be performed on $\boldsymbol{A}$: the optimal value of β is chosen by minimizing the average rank of the newly added relevant images and of the original ones if they were selected from the queried databases. As in the complete representation the query images obtained from the queried databases always appear in the first positions (having a zero distance), this procedure is only useful to optimize the condensed query. However, the value of β can also be optimized for the complete representation in the following way: for each image in the query set obtained from the queried databases, a synthetic query is created by removing it and its rank in the resulting system answer stored. The average rank over the synthetic queries is then used for the optimization (see Fig. 7). The condensed query representation is then employed using the lowest value of β for which the rank correlation of the condensed and complete representation results satisfy the required confidence level.

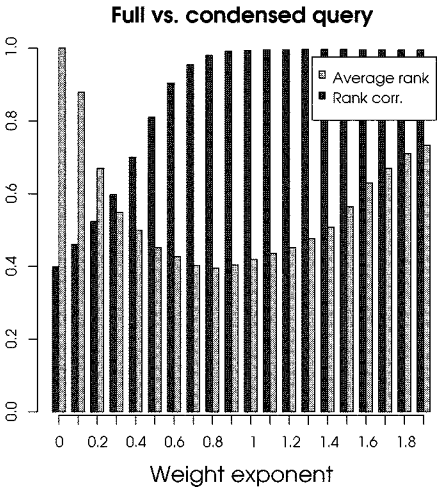

Fig. 7. Average rank of the query images when a condensed search is employed with different weights. The rank correlation with the results obtained using all the images in the query bag is also reported.

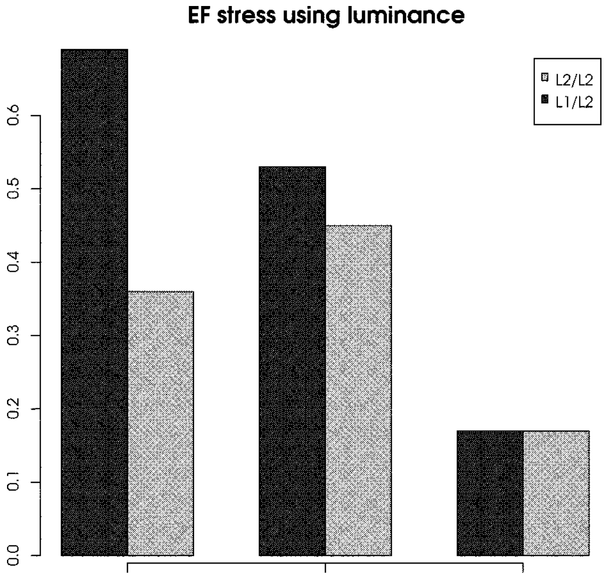

Fig. 8. Final stress J_{ef} for different distances and dimensionality reduction techniques.

VI. Database Organization and Browsing

Browsing an image database is substantially different from querying it and presents specific interaction problems. These problems become more evident when multiple databases are browsed simultaneously. As large image databases cannot be

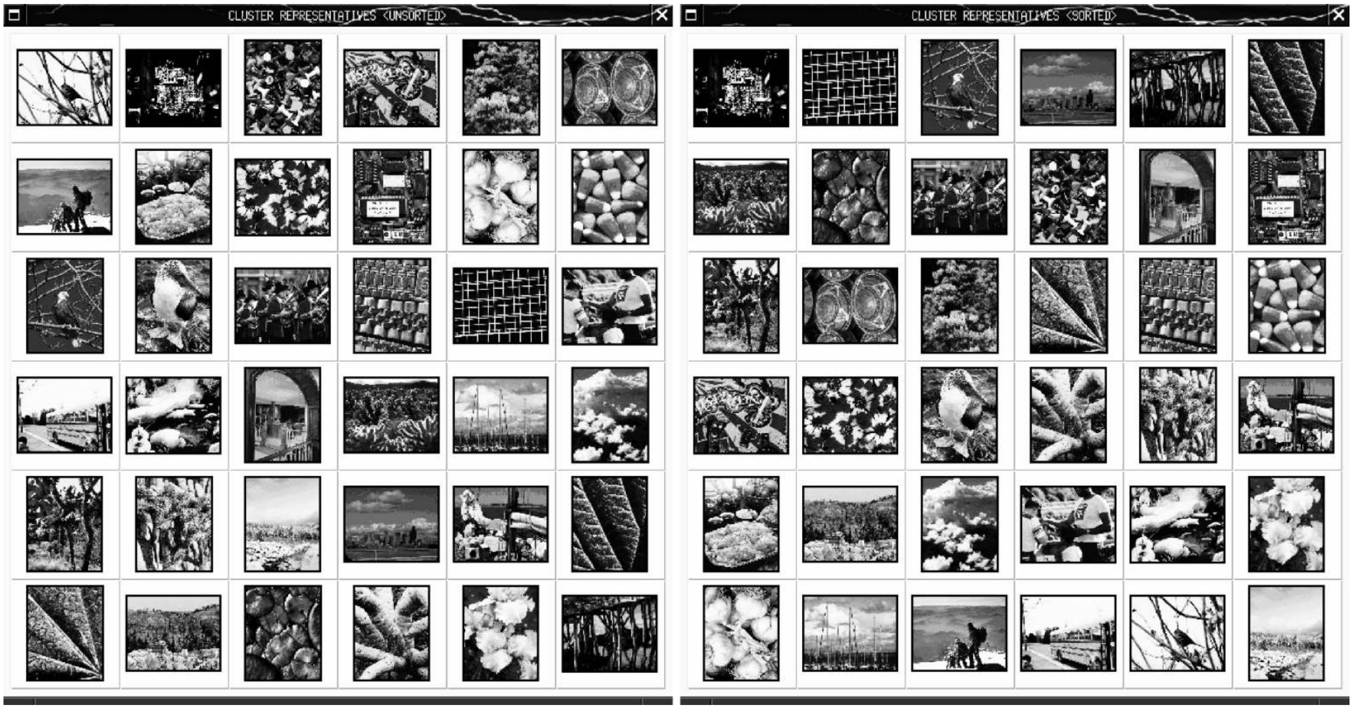
CLUSTER REPRESENTATIVES <UNSORTED>
CLUSTER REPRESENTATIVES <SORTED>

Fig. 9. Effect of *sorting* the cluster representatives of two databases using global luminance information. Note how the picture to the right, with sorted representatives, appears more homogeneous and easy to analyze than the one to the left where images are unsorted.

presented in their entirety to the user, useful abstractions should be developed, presenting to the user a limited number of key images which can be used to pivot the search. It would also be important to present the images in such a way that their layout reflect the notion of similarity used by the system: nearby images should be visually similar and similarity relations among images should be preserved as much as possible. The task is further complicated by the dynamic nature of image similarity due to the adoption of relevance feedback techniques which change the metric structure of the descriptor space.

The solution adopted by COMPASS relies on clustering and multidimensional scaling techniques. Each image database is clustered into groups of images similar to each other according to the L_1 norm. Each group is then represented by a key image, the image closest to the cluster center. The set of cluster representatives provides the required abstraction for database browsing. Each representative acts as an hyper-link to the complete cluster population that can be shown to the user on demand.

As the number of clusters is usually much smaller than the number of images in the database, computationally expensive algorithms can be applied to organize the visual presentation of the key images to reduce browsing stress. Let us consider the situation in which two databases are used for browsing and a weighted L_1 norm is used for comparing the images. Key images should be arranged on the screen in such a way that nearby images are visually similar. The user can choose one (or more) of the image features, e.g., luminance, as a sorting key for the arrangement of the images onto the screen. The key images from the two databases are then arranged onto a line preserving as far as possible their mutual similarities as quantified by L_1 distance. This can be accomplished by using multidimensional scaling techniques [16] which deal with the following problem:

Given a set of similarities or distances between every pair of N items, find a representation of the items in few dimensions such that the inter item proximities nearly match *the original similarities or distances.*

While it is not necessary to use the values of the similarity between the samples (the rank orders could be used instead), the following discussion is restricted to the case where the values are explicitly used. Following [16], let $\boldsymbol{X} = \{\boldsymbol{x}_i\}_{i=1,\cdots,n}$ be a set of samples represented by points in $\boldsymbol{R}^K$. Let $\boldsymbol{Y} = \{\boldsymbol{y}_i\}_{i=1,\cdots,n}$ be the projection of $\boldsymbol{X}$ in $\boldsymbol{R}^k$, $\boldsymbol{k} < \boldsymbol{K}$ and δ_{ij}, d_{ij} the distances between samples i and j in $\boldsymbol{R}^K$ and $\boldsymbol{R}^k$ respectively. The objective of multidimensional scaling is then to find a configuration of image points $\{\boldsymbol{y}_i\}_{i=1,\cdots,n}$ for which $\delta_{ij} \approx d_{ij}$. A measure of *closeness,* usually called *stress,* must then be introduced: the lower the stress the better the obtained scaling. A commonly used measure of stress is

$$J_{ef} = \left(\sum_{i<j} \delta_{ij}\right)^{-1} \sum_{i<j} (d_{ij} - \delta_{ij})^2 \delta_{ij}^{-1} \tag{11}$$

It is important to observe that it is not necessary for the distances d_{ij} and δ_{ij} to be computed in the same way, e.g., using the Euclidean distance. This is true in particular in the reported context as the original distances between image descriptors are computed using the weighted L_1 norm while the distances of the projected set $\boldsymbol{Y}$ are computed using the Euclidean norm which corresponds to the user perceived distance in the space where images are to be presented. The necessity of relying on different metric structures in the original and projected spaces somehow limits the choice of the algorithms to be used in finding the desired configuration. For instance, a commonly used technique to project vectors onto a lower dimensional space is given by the Principal Component Analysis (PCA). This algorithm provides good results when the metric structure in the original and reduced space is given by the Euclidean norm. Direct minimization of the stress value leads to the so called Sammon mapping which does not depend on the type of distances used in the source and destination spaces but suffers from the following disadvantages:

- high computational requirements;
- presence of many suboptimal local minima;
- the map is given as a look-up table that must be recomputed whenever new points are added.

More recently, a fast algorithm for multidimensional scaling, FastMap, was introduced [22]. Given the set of original distances, the algorithm finds a representation of the original points in $\boldsymbol{R}^k$ computing the set δ_{ij} using the Euclidean distance. When new data points are added, they can be easily projected in the reduced space.

In the presented work the three cited algorithms (PCA, Sammon mapping, and FastMap) have been compared in the task of projecting points from $\boldsymbol{R}^{16}$ to $\boldsymbol{R}^1$ using J_{ef} as stress indicator and different metrics in the source and destination spaces. Points in source space are given by the image luminance histograms. The results are reported in Fig. 8. Sammon mapping outperforms the competing algorithms in quality and is considered to be a viable choice in spite of its shortcomings. An example of the application of multidimensional scaling to the presentation of images using as the luminance histogram as derived descriptor is reported in Fig. 9.

VII. Conclusions

In this paper an architecture for a general image retrieval and browsing system featuring relevance feedback was presented and discussed. In particular, the possibility of tuning search strategies and comparison metrics to varying user behavior was investigated and novel solutions presented using pattern analysis techniques. The resulting image retrieval system is able to optimize retrieval speed by reducing the number of query images while preserving retrieval effectiveness. The use of local modification of the image comparison metric, coupled to the use of negative examples further enhances the ability of the system at modeling user needs on a per query basis. The possibilities offered by data analysis techniques have been adapted to the activity of database browsing, suggesting how clustering techniques and multidimensional scaling can be used to present a database map to the user.

References

[1] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin, "The QBIC Project: Querying images by content using color, texture, and shape," *Proc. SPIE*, pp. 173–181, 1993.

[2] J. R. Smith and S. F. Chang, "Quad-tree segmentation for texture-based image query," in *Proc. ACM Multimedia'94 Conf.*, Oct. 1994.

[3] ——, "VisualSEEk: A fully automated content-based image query system," in *Proc. ACM Multimedia'96 Conf*, Nov. 1996.

[4] P. Kelly and M. Cannon, "Query by image example: The CANDID approach," *Proc. SPIE*, vol. 2420, pp. 238–248, 1995.

[5] S. F. Chang, J. R. Smith, H. J. Meng, H. Wang, and D. Zhong, "Finding images/video in large archives," *D-Lib Mag.*, Feb. 1997.

[6] S. Mehrotra, Y. Rui, M. Ortega, and T. S. Huang, "Supporting content-based queries over image in Mars," in *Proc. 4th IEEE Int. Conf. Multimedia Computing and Systems*, Ottawa, ON, Canada, June 3–6, 1997, pp. 623–633.

[7] R. Brunelli and O. Mich, "On the use of histograms for image retrieval," in *Proc. ICMCS'99*, Florence, June 1999.

[8] I. J. Cox, M. L. Miller, S. M. Omohundro, and P. N. Yianilos, "PicHunter: Bayesian relevance feedback for image retrieval," in *Proc. Int. Conf. Pattern Recognition*, Austria, 1996.

[9] Y. Ishikawa, R. Subramanya, and C. Faloutsos, "MindReader: Querying databases through multiple examples," in *Proc. Int. Conf. Very Large Data Bases*, 1998.

[10] S. Sclaroff, L. Taycher, and M. La Cascia, "ImageRover: A content-based image browser for the world wide web," in *Proc. IEEE Workshop on Content-based Access of Image and Video Libraries*, 1997.

[11] K. Porkaew, K. Chakrabarti, and S. Mehrotra, "Query refinement for multimedia similarity retrieval in MARS," in *Proc. ACM Int. Multimedia Conference*, Orlando, FL, Nov. 1999.

[12] Y. Rui, T. S. Huang, and S. Mehrotra, "Content-based image retrieval with relevance feedback in Mars," in *Proc. IEEE Int. Conf. Image Processing'97*, Oct. 1997, pp. 815–818.

[13] M. Ortega, Y. Rui, K. Chakrabarti, K. Porkaew, S. Mehrotra, and T. S. Huang, "Supporting ranked Boolean similarity queries in MARS," *IEEE Trans. Knowl. Data Eng.*, vol. 10, Dec. 1988.

[14] G. W. Milligan and M. C. Cooper, "An examination of procedures for determining the number of clusters in a data set," *Psychometrika*, vol. 50, pp. 159–179, 1985.

[15] A. K. Jain and J. V. Moreau, "Bootstrap technique in cluster analysis," *Pattern Recognit.*, vol. 20, no. 5, pp. 547–568, 1987.

[16] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

[17] R. C. Dubes, "How many clusters are best?—An experiment," *Pattern Recognit.*, vol. 6, no. 20, pp. 645–663, 1987.

[18] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. COM-28, no. 1, pp. 84–95, 1980.

[19] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data. An Introduction to Cluster Analysis*. New York: Wiley, 1990.

[20] Y. Rui, T. S. Huang, S. Mehrotra, and M. Ortega, "A relevance feedback architecture in content-based multimedia information," in *Proc. IEEE Workshop on Content-Based Access of Image and Video Libraries*, Puerto Rico, June 1997, pp. 82–89.

[21] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C, Second Edition*. Cambridge, U.K.: Cambridge Univ. Press, 1988.

[22] C. Faloutsos and K. I. Lin, "FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets," in *Proc. SIGMOD'95*, 1995, pp. 163–174.

Roberto Brunelli was born in Trento, Italy, in 1961. He received the degree in physics (with honors) from the University of Trento in 1986.

He joined ITC-irst, Povo, Italy, in 1987, where he works in the Computer Vision Group of the Interactive Sensory Systems Division. In the past, he was involved in research on computer vision tools, analysis of aerial images, optimization, and face analysis. His current major involvement is in the area of image similarity retrieval and multimedia databases.

Ornella Mich was born in Tesero, Trento, Italy, in 1959. She received the degree in electronic engineering from the University of Padova, Italy, in 1987.

She then worked on radar electronics for jet-fighters. After teaching electrotechnics in a high school, she joined ITC-irst, Povo, Italy, in 1989, collaborating on VLSI research. Since 1990, she has been working in the field of computer vision in the Interactive Sensory Systems Division.