MuSIQUE: A Multi-System Image Querying User Interface

Ilaria Bartolini and Paolo Ciaccia

DEIS - IEIIT-BO/CNR, University of Bologna,
Viale Risorgimento, 2 - 40136 Bologna, Italy
{ibartolini, pciaccia}@deis.unibo.it

Abstract. Current image retrieval systems do not provide users with adequate interfaces able to reduce the *semantic gap* existing between the high-level semantics of images (as perceived by the users) and their low-level machine representation. In this paper, we present MuSIQUE, a *multi-system* user interface, mainly designed for experimental purposes, that makes it possible to compare different systems into an integrated environment. By critically reviewing limits of the interfaces of state-of-the-art image retrieval systems, we have been able to integrate into MuSIQUE all the advanced functionalities of such systems and, at the same time, to overcome their shortcomings. A detailed description of the interface features of MuSIQUE, concerning querying, visualization, and browsing, is reported. Moreover, a brief example of MuSIQUE in action is shown, aiming to illustrate the effectiveness of the proposed facilities.

1 Introduction

The wide diffusion of images on the web and in many real world applications has raised the necessity to provide an effective support to search and browse large image collections. Since manual annotation aiming to describe images' contents is impracticable for large image databases, even because of the inherent subjectivity that is present in the annotation process,[1] content-based image retrieval (CBIR) systems, which automatically extract from each image a set of relevant low-level *features* (such as color distribution, texture, shape descriptors and objects' location information) have become extremely popular in recent years [8, 14]. Although early CBIR systems characterized each image with a single vector of *global features*, this approach is nowadays recognized to yield poor performance when compared to that adopted by *region-based* systems [18, 5, 2, 13, 20], which share the idea of automatically segmenting the image into a set of coherent regions and then of describing each of such regions using a vector of *local features*. This results in a more precise description of the image content, especially when images have a complex and heterogeneous structure.

In spite of the many produced efforts, CBIR systems are still far from having overcome the *semantic gap* between the low-level machine representation of

[1] See http://elib.cs.berkeley.edu/photos/blobworld/warning.html.

images and the high-level concepts that humans associate to images [16]. Indeed, all existing CBIR systems lack some basic facility that could potentially lead to alleviate the hard-to-solve semantic gap problem. Some systems (e.g., [5, 13]) allow the user to execute queries by selecting the region(s) of an image of interest and then return a set of best-matching images, each accompanied by its segmented version where regions matching those of the query are highlighted. In this way, the user gets a better cue on how the underlying system is working, i.e. on what makes images in the result most similar to the query image, yet the flat, list-based, display of results does not provide any information about mutual similarities between result images and about the possible presence of clusters among them. This, together with the absence of advanced browsing facilities to explore the result set, severely limits the user in understanding how images are organized in the feature space and, consequently, reduces the chances to find the images matching the concepts she has in mind. To the best of our knowledge, the only effort that tries to combine advanced querying facilities with more meaningful visualization techniques is the one described in [15]. However, the system in [15] is still based on global features, thus missing all the advantages offered by the region-based approach.

In this paper, we will focus on the possibility of reducing the semantic gap between the user and the system by proposing, essentially for experimental intents, an extensible user interface that brings together the functionalities offered by the more advanced region-based image retrieval tools [5, 13, 20]. Our MuSIQUE (*MUlti-System Image Querying User interfacE*) web-based interface is based on a *multi-system* architecture allowing the user to choose which image retrieval system to use for searching. The user can then decide to query the dataset by specifying the regions of the query image of interest and visualize the result set in the most profitable way to interpret the result itself (this is the case when the user has already in mind the image or the class of images she is looking for). On the other hand, when the user does not know yet what she is looking for, but is just interested in exploring the content of the image set, advanced browsing facilities are provided.

The paper is organized as follows. In Section 2 we investigate the functionalities of the user interfaces of state-of-the-art content-based image retrieval systems and discuss their pros and cons. In Section 3 we introduce MuSIQUE, illustrating its functionalities that include all the positive aspects of existing interfaces and add some new useful skills able to overcome negative aspects, and in Section 4 we describe its *multi-system* architecture. In Section 5 we show MuSIQUE in action, showing search results obtained using the Windsurf region-based image retrieval system [2, 4]. To prove the effectiveness of the multi-system functionality, we also show a comparison between the Windsurf and Blobworld [5] systems. Finally, Section 6 concludes the paper, drafting possible directions for future work.

2 User Interfaces for Content-Based Image Retrieval

In the following, we analyze the basic functionalities offered by the user interfaces of existing CBIR systems, restricting the attention to region-based ones because of their inherent superiority, as already discussed in the introduction. For convenience, functionalities are grouped into the following three categories:

Querying This includes all those facilities that allow the user to formulate a query.

Visualization This category concerns aspects related to how the result of a query is displayed to the user.

Browsing Browsing facilities should complement those of querying and are particularly important when the user is not looking for specific images and just wants to explore the image collection, and also if she wants to "look around" the result of a query.

2.1 Querying

There are basically two ways for querying a CBIR system: *Query By Example* (QBE), where the user provides a query image and decides which part of the image is relevant for the search, and *Query By Sketch* (QBS), where the user directly sketches the query objects using supplied drawing facilities.

In the QBE case, the set of images available to the user for choosing the query can vary. For example, with SIMPLIcity [20] the user can choose between a fixed set of images in the database, select a random image of the database, or provide the URL of an image not included in the database. However, even if SIMPLIcity characterizes each image as a set of regions, its interface does not allow the user to select the regions of interest in the image. The NeTra [13] system, on the other hand, provides a semantic classification of the image dataset: By selecting a particular category, the list of the images belonging to the class is shown, and the user can select the query among them. The web interface provided by the Blobword system [5], like SIMPLIcity, presents to the user a set of representative images to choose from. Both Blobworld and NeTra interfaces allow the user to select which parts of the query to use during the search, even if both systems limit such choice to just one region. Blobworld also allows to assign weights to the region features (i.e. color, texture, and shape).

For the QBS modality, only SIMPLIcity provides opportune drawing facilities that allow the user to manually define the query image.

2.2 Visualization

The traditional way to show the result of a query consists in displaying the set of images in a two-dimensional grid in decreasing order of similarity (we refer to this method as *flat visualization*). SIMPLIcity, NeTra, and Blobworld all provide this visualization modality. Moreover, NeTra and Blobworld return, along with each image, the corresponding segmented image, highlighting which image regions match the query ones. Flat visualization has two major drawbacks. When the

user has only a vague idea of what she is looking for, it is important that related images in the result set are grouped together, which is not the case with flat visualization (this is because result images are linearly ranked with respect to the query image, and their mutual similarities are ignored to this end). The second problem plaguing flat visualization is that it is hard to get a global view of the result images. Clearly, the solution of reducing the cardinality of the result set can alleviate this problem, yet it increases the chance of not obtaining any relevant result. To obviate above problems, a *spatial visualization* approach can be adopted. In this case, images are represented on the plane of the screen with similar images grouped together rather then being scattered along the entire result list. With this type of visualization, the user can see relations between the images, and refine her query in a more natural way. Unfortunately, none of the above mentioned CBIR systems provides a spatial visualization facility. A remarkable exception is the image retrieval tool developed at the University of Stanford [15]; however, since it is not based on image segmentation, it suffers all the drawbacks of the global approach to CBIR.

2.3 Browsing

When the user is not interested in looking for a particular image but just in exploring the content of the image database, navigation facilities become very important. Browsing skills of traditional CBIR systems are usually limited to displaying a small set of images at a time (we refer to such modality as *flat browsing*). The main problem with flat browsing is that, since the user has only a global snapshot of the whole database, only indication about where to go next are suggested, thus the browsing process becomes disorienting for the user: Too few images are displayed at a time and at the end of the exploration it is difficult for the user to remember about the first shown images. Like in the case of visualization, also here spatial visualization can be helpful. In this case the user can see larger portions of the database at a glance. Once she points to the image of interest on the display, the system zooms in and finds more images that are similar to that image. By iterating this refinement process, the user can quickly reach the relevant parts of the database (we refer to this second modality as *spatial browsing*). Among the mentioned CBIR systems, only SIMPLIcity and NeTra offer a flat browsing and none of them can provide the spatial browsing functionality.

3 MuSIQUE

In this section we present MuSIQUE, an advanced web interface that can be viewed as the convergence point of the querying, visualization, and browsing facilities described in Section 2 (the interface layout is reported in Figure 1).

We designed MuSIQUE as a *multi-system* web-based interface for *region-based* image retrieval systems, in order to give the user the opportunity to select both the system to be used and the image query, as well as the image region(s) of interest. The MuSIQUE architecture is detailed in Section 4.

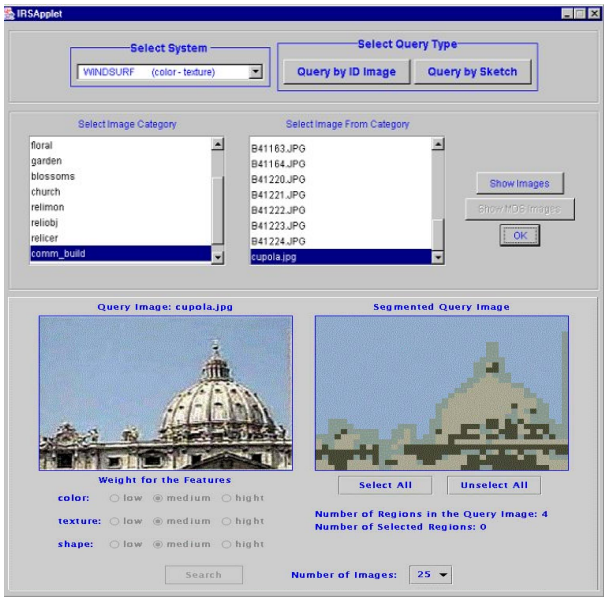


Fig. 1. The MuSIQUE query interface.

We supply our web interface with both QBE and QBS facilities. The user can choose the preferred modality by just pressing the corresponding button provided by the interface (see top-right corner in Figure 1). When the user submits a query, MuSIQUE "sends" the query to the chosen search engine and then shows the result list using the flat visualization modality. The user can then choose to look at the results using the spatial visualization. MuSIQUE also allows the user to browse the image collection choosing a flat or a spatial browsing facility.

In the following, we provide a detailed description of MuSIQUE features.

Querying For QBE ("Query by ID Image" button in Figure 1), MuSIQUE presents a set of string categories for choosing the query image (for example, in Figure 1 the selected category is "comm_build"). By selecting the name of a category, a list of images belonging to that category is displayed, from which the user can select the query. Obviously, such modality is only useful for those users that already know the content of the dataset. MuSIQUE also provides a "Show Images" button that enables the user to display the images belonging to the selected category and to choose the query by clicking on it. Once the query image has been selected (this is the "dome of St. Peter" in Rome in Figure 1), the interface also displays the corresponding segmented image (i.e. a representation of its regions), so that the user can select a subset of the query regions to be used in order to formulate the query in a more precise way. The user can also assign weights to the features (color, texture, and shape) characterizing each region, thus actually changing the distance function used to compute images'

dissimilarities. Finally, the user selects the number of images to be returned in the result set by way of the provided combo box and presses the "OK" button to execute the query.

For the QBS modality, MuSIQUE provides the user with a set of common drawing functionalities that enable her to directly draw the image of interest. We do not detail this further for lack of space.

Visualization MuSIQUE provides a default flat visualization of the query result. To give the user the opportunity of personalizing the layout of the display, MuSIQUE allows the user to choose the size used to display the images. In fact, we believe that, as is the case for fonts sizes in word processors, also in the image retrieval context it is very important to allow choosing the image size that best fits user needs. Then, to benefit of a more coherent view of the query results, the user can obtain a spatial display of the result set by clicking the "Space Visualization" button in the result panel. In this case the 2-D layout of result images depends on their mutual dissimilarities, which in turn depend on user-assigned weights. In this way, spatial visualization can also help to better understand how weights actually modify the "feature space" in which images are represented. An example of all the visualization facilities is reported in Figures 3 and 4 for the "dome of St. Peter" query.

Moreover, with both visualization modalities, the user can select one image of interest on the display to formulate a new (possibly more specific) query. A new set of images will be returned and displayed by the system. By iterating this process, the user can thus refine the result set obtaining images closer and closer to what she is looking for.

Browsing The flat browsing facility is represented by the possibility to display the content of the database by starting from the snapshot related to a specific category of the dataset, and going back and forth from this display (see Figure 5). Concerning spatial browsing, MuSIQUE supplies a specific button that allows the user to see at a glance the content of the database at different hierarchical levels. Once the user finds an image of interest on the display at some level, by clicking on such button a zooming of the display is obtained, and more images similar to the pointed image are obtained. By iterating this process, the user is able to quickly navigate the portion of the image space of interest (for an example, see Figure 6). To achieve this, a hierarchical clustering structure is built on the image database and images displayed at each level correspond to the centroids of clusters at such level. In our current implementation, the clustering is obtained through an index structure (in the specific case, an M-tree [6]) built on the dataset and the hierarchy of clusters corresponds to the levels of the tree.

One common problem with both the visualization and browsing facilities consists in the possible overlapping of images in the spatial display. This happens because the distance between many of the images in the two-dimensional configuration is lower than their size. To avoid such problem, MuSIQUE allows the user to change the images' sizes together with the size of the display panel. For an example of such facility, see Figures 4 and 6.

3.1 Multidimensional Scaling

The general problem related to the spatial display consists in how to represent complex objects (e.g., images, DNA sequences, documents, etc.) in a low dimensional space (commonly 2-D or 3-D) preserving the mutual distances between objects [7]. For the specific case of images, by using the dissimilarities between all pairs of images we would like to obtain an embedding in a 2-D space, in order to place the images on the screen so that distances on the screen reflect as closely as possible dissimilarities between images. To achieve this goal, we used the *Multidimensional Scaling* (MDS) [19, 17, 12] technique. MDS can be defined in several ways, some of which even allow for non-metric distances (i.e. where the triangle inequality is not satisfied), but in all cases the objective is the same and can be formalized as follows [9].

Problem 1. Given a set of N objects together with their dissimilarities $d_{i,j}$, compute a configuration of points o_i in a low-dimensional space R^D (e.g., D=2) so that the Euclidean distances $d'_{i,j}$ between the points in the obtained R^D space match the original dissimilarities $d_{i,j}$ as close as possible. □

One of the more common formalization of the problem is due to Kruskal [12] and requires minimizing a so-called *stress* function:

$$stress = \left(\frac{\sum_{i,j} (d_{i,j} - d'_{i,j})^2}{\sum_{i,j} d_{i,j}^2} \right)^{1/2} \tag{1}$$

The stress is a non negative number that indicates how well distances are preserved by the embedding. Minimizing stress is a non-linear problem, where the variables are the $N \cdot D$ coordinate values corresponding to the embedding. It has to be noted that MDS may not obtain the minimum stress, reporting instead an embedding achieving a local minimum [9]. In our current implementation we considered the Torgerson MDS algorithm [19].

MDS techniques are not the only solution: A simple alternative is represented by FastMap [7]. FastMap is a technique based on concepts from linear algebra that can therefore work properly only with Euclidean spaces (i.e. when the distance function d is the Euclidean metric). In fact, when the original space is not (isometric to) an Euclidean space, the mapping produced by FastMap may introduce considerable distortion [9], thus leading to high stress values. Even if, from a computational point of view, FastMap is cheaper than MDS (linear vs quadratic complexity), we decided to use MDS, since in our context no assumption can be made on the dissimilarity function used by underlying CBIR systems. It has also to be noted that in MuSIQUE spatial visualization only concerns a limited number of images (the result set of a query or the content of an index node), thus $O(N^2)$ complexity is not a major issue.

4 The Architecture

The MuSIQUE architecture is reported in Figure 2. As the figure shows, one

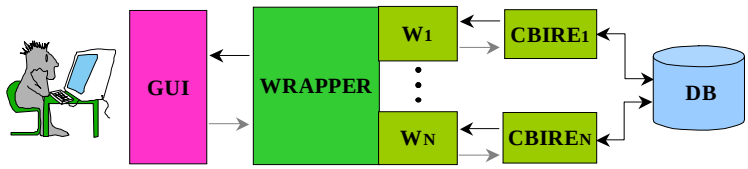


Fig. 2. The MuSIQUE architecture.

of the main characteristic of our web interface is its *multi-system* facility. This feature is very important because it allows to perform the same set of queries on different CBIR systems and to compare the results in an extremely simple way (see Figure 7 for an example). Indeed, nowadays the only way to compare different systems is to try each system separately, and then make comparisons on the result sets. Most of the times, however, each system works with its own image database and, consequently, a fair comparison becomes impossible.

In details, through the graphical user interface (GUI) shown in Figure 1, the user selects the CBIR system she wants to use for searching, together with the image query. Then, the global wrapper receives the user request and, by calling the specific CBIR wrapper, sends the request to the CBIR engine. The CBIR engine, interacting with image database, satisfies the request and returns the result list to its wrapper and, by the global wrapper, to the GUI. Images belonging to the returned set are then displayed to the user through the GUI.

Note that wrappers for other systems can be added to MuSIQUE at any time, by just implementing the specific wrapping methods and supplying the corresponding CBIR engine library.

Technical details related to the client and server side of the MuSIQUE web interface are as follows.

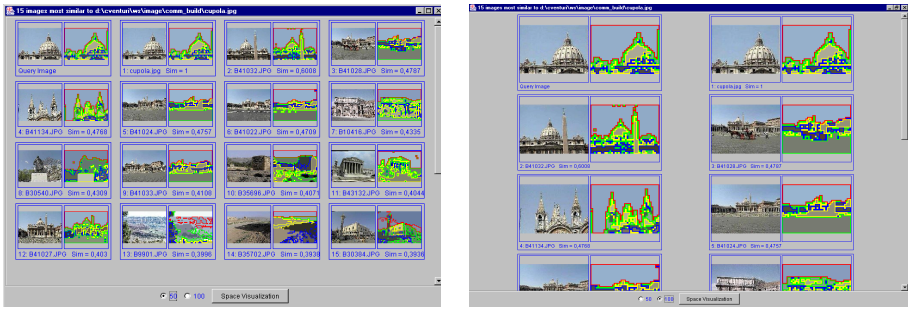
Client For the current implementation of the MuSIQUE client side, we used the *Applet Java* technology (J2SE, v. 1.4). This choice allows to combine the expressive power of a programming language like *Java* with the possibility to include some code inside the web pages and execute it within the web browser running on a client device.

Server The current implementation of the server side of MuSIQUE uses *Java Servlets* (v. 2.2) [10] and *Jakarta Tomcat* (v. 1.3) technologies [1]. Java servlets offer a fast, powerful, portable replacement for *Common Gateway Interface* (CGI) scripts [11] and can be seen as the counterpart of the applet server side. CGI scripts are still popular, but by now Java Servlets represent an optimal solution to the main CGI problems. Unlike CGI, where for each request a new process is created, the Java Servlets technology allows to run each servlet in a separate thread, eliminating the overhead due to the fork() and exec() operations proper of CGI. Although today there are many web servers able to support such technology, we chose to use Jakarta Tomcat, the reference implementation for Java Servlets v. 2.2.

5 Playing MuSIQUE

To show MuSIQUE in action, we implemented some CBIR software engines in our architecture. In particular, we used WINDSURF, the region-based image retrieval system we proposed in [2, 4], and we also implemented the wrapper and the software engine for Blobworld [5]. In the following, we present a brief example of a query and browsing using the MuSIQUE interface with the WINDSURF system. A comparison of the results obtained by WINDSURF and Blobworld for the same query is also provided. Experimentation has been performed on a dataset consisting of about 10000 real-life images extracted from a CD-ROM of *IMSI-PHOTOS*.[2]

Figure 3 shows the result for the "dome of St. Peter" query (the top-left image in the figure) using the flat visualization. In the specific example, the user selected all the regions of the image as relevant for the search. In particular, Figure 3 (a) reports a display for a more general view of the result set (images are displayed at the 50% of their original size). Figure 3 (b), instead, represents less images on the screen but with a higher detail level (images are shown using their original size). The result of the same query is reported in Figure 4 using the spatial visualization. In details, Figure 4 (a) displays the query result using



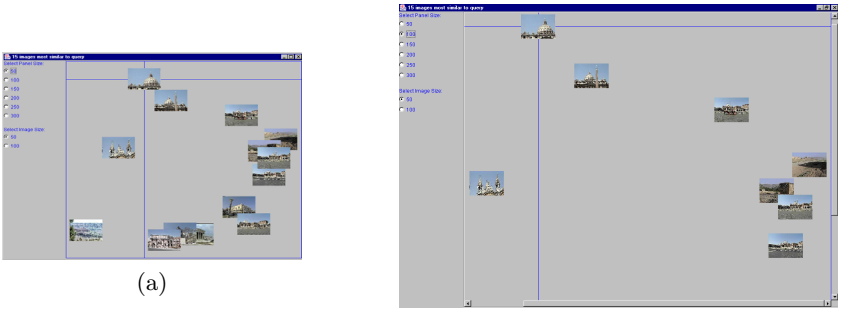
(a) (b)

Fig. 3. Flat visualization with image size set to 50% (a) and to 100% (b).

the default panel size (equal to the 50% of the original panel size), while Figure 4 (b) shows the same images using the 100% of the panel size, so as to avoid the overlap problem related to the spatial visualization.

In Figures 5 and 6 a browsing example of the image database is reported using the flat and the spatial modality, respectively. In the flat browsing example (Figure 5) the user starts the database exploration from the semantic category "landscape" and, from this point, she can go back and forth in navigating the dataset by means of the buttons on the display panel. The spatial example (Figure 6), shows in the left panel a picture of the database at some level of detail,

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



(a) (b)

Fig. 4. Spatial visualization with panel size set to 50% (a) and to 100% (b).

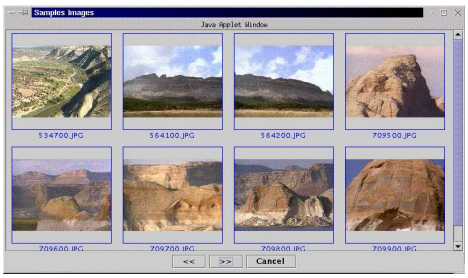


Fig. 5. Flat browsing example.

and in the right panel, the set of images which are "close" to the "landscape" image on which the user clicked on (i.e. the image with a red frame in the first panel). This way, an useful and clear visualization of the relations between the displayed images is provided to the user.

Finally, Figure 7 demonstrates the importance of the multi-system property of MuSIQUE comparing the result sets obtained by WINDSURF and Blobworld for the "flowers" query (the top-left image in the figure).

6 Final Discussion

In this work we have considered the problem of how to improve the quality of interaction between users and CBIR systems so as to reduce the "semantic gap" existing between the high-level semantics of images and their low-level machine representation. Starting from existing user interfaces of modern CBIR systems, we have designed and implemented MuSIQUE, an advanced user interface based on a multi-system architecture that allows the comparison between different systems for experimental intents. MuSIQUE provides advanced query facilities enabling the user either to select an image from the dataset (by choosing the portions of image of interest), or to depict the requested image by way of specific drawing tools. Then, the result of each query can be displayed using both flat

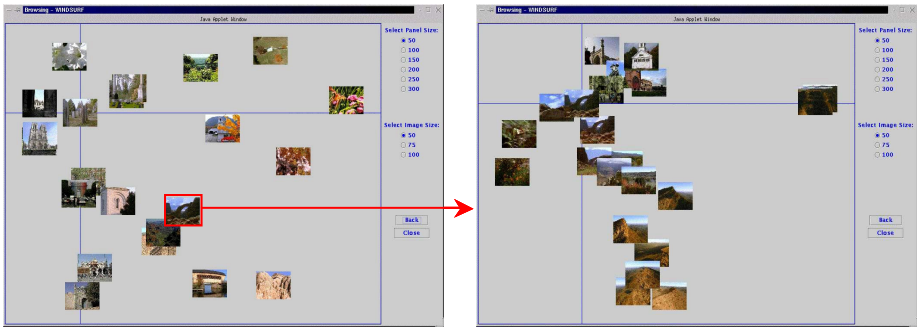


Fig. 6. Spatial browsing example.

(a) (b)

Fig. 7. Results of the "Flowers" query: WINDSURF (a) and Blobworld (b).

and spatial visualization modalities. Moreover, the user can customize the layout of the result displays. Finally, to allow the user to navigate the image database, flat and spatial browsing modalities are provided.

Considering the present implementation of MUSIQUE a number of new facilities could be added. First, it would be useful to have MUSIQUE accept user preferences over the result images, thus allowing the user to provide relevance judgments, and possibly to "learn" them [3]. Using the acquired knowledge about the user feedback, MUSIQUE could improve the quality of each user search. A further challenge is represented by the possibility to add to the MUSIQUE architecture a module able to integrate the results obtained by different systems for the same set of queries. In this way, in fact, it would be possible to obtain the most relevant set of images by "merging" the best results obtained by each system.

References

1. The Apache Jakarta project. URL: `http://jakarta.apache.org`.

2. S. Ardizzoni, I. Bartolini, and M. Patella. Windsurf: Region-based image retrieval using wavelets. In *Proc. of the 1st Int'l Workshop on Similarity Search*, pages 167–173, Florence, Italy, Sept. 1999.
3. I. Bartolini, P. Ciaccia, and F. Waas. FeedbackBypass: A new approach to interactive similarity query processing. In *Proc. of the 27th Int'l Conf. on Very Large Data Bases*, pages 201–210, Rome, Italy, Sept. 2001.
4. I. Bartolini and M. Patella. Correct and efficient evaluation of region-based image search. In *Atti dell'Ottavo Convegno Nazionale SEBD*, pages 289–302, L'Aquila, Italy, June 2000.
5. C. Carson, M. Thomas, S. Belongie, J. M. Hellerstein, and J. Malik. Blobworld: A system for region-based image indexing and retrieval. In *Proc. of the 3rd Int'l Conf. on Visual Information Systems*, pages 509–516, Amsterdam, The Netherlands, June 1999.
6. P. Ciaccia, M. Patella, and P. Zezula. M-tree: An efficient access method for similarity search in metric spaces. In *Proc. of the 23rd Int'l Conf. on Very Large Data Bases*, pages 426–435, Athens, Greece, Aug. 1997.
7. C. Faloutsos and K.-I. Lin. FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *Proc. of the 1995 ACM SIGMOD Int'l Conf. on Management of Data*, pages 163–174, San Jose, CA, June 1995.
8. M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video content: The QBIC system. *IEEE Computer*, 28(9):23–32, Sept. 1995.
9. G. R. Hjaltason and H. Samet. Contractive embedding methods for similarity searching in metric spaces. Technical Report CS-TR-4102, Computer Science Department, University of Maryland, College Park, MD, Feb. 2000.
10. J. Hunter and W. Crawford. *Java Servlet Programming*. O'Reilly, 1998.
11. A. Korthaus and S. Kuhlins. Java servlets versus CGI - implications for remote data analysis. In *Proc. of 23rd annual Conf. on Classification and Information Processing at the Turn of the Millenium*, pages 227–236, Heidelberg, Germany, Mar. 2000.
12. J. B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesys. *Psychometrika*, 29(1):1–27, Mar. 1964.
13. W.-Y. Ma and B. S. Manjunath. NeTra: A toolbox for navigating large image databases. *Multimedia Systems*, 7(3):184–198, May 1999.
14. A. Pentland, R. W. Picard, and S. Sclaroff. Photobook: Content-based manipulation of image databases. In B. Furht, editor, *Multimedia Tools and Applications*, chapter 2, pages 43–80. Kluwer Academic, 1996.
15. Y. Rubner and C. Tomasi. *Perceptual Metrics for Image Database Navigation*. Kluwer Academic Publishers, Boston, MA, Dec. 2000.
16. S. Santini and R. Jain. Integrated browsing and querying for image databases. *IEEE Multimedia*, 7(3):26–39, July 2000.
17. R. N. Shepard. The analysis of proximities: Multidimensional scaling with an unknown distance function. *Psychometrika*, 27(2):125–140, 1962.
18. J. R. Smith and S.-F. Chang. VisualSEEk: A fully automated content-based image query system. In *Proc. of the 4th ACM Int'l Conf. on Multimedia*, pages 87–98, Boston, MA, Nov. 1996.
19. W. S. Torgerson. *Theory and Methods of Scaling*. J. Wiley, New York, NY, 1958.
20. J. Z. Wang, J. Li, and G. Wiederhold. SIMPLIcity: Semantics-sensitive Integrated Matching for Picture LIbraries. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(9):947–963, Sept. 2001.