Interface for visualization of image database in adaptive image retrieval systems (AIRS)

Anca Doloc-Mihu, Vijay V. Raghavan, Surendra Karnatapu and Chee-Hung Henry Chu

The Center of Advanced Computer Studies, University of Louisiana at Lafayette, USA

ABSTRACT

In an Adaptive Image Retrieval System (AIRS) the user-system interaction is built through an interface that allows the relevance feedback process to take place. Most existing image retrieval systems simply display the result list of images (or their thumbnails) to the user in a 2D grid, without including any information about the relationships between images. In this context, we propose a new interactive multiple views interface for our AIRS, in which each view illustrates these relations by using visual attributes (colors, shapes, proximities).

We identify two types of users for an AIRS: a user who seeks images whom we refer to as an end-user, and a user who designs and researches the collection and the retrieval systems whom we refer to as a researcher-user. With such views, the interface allows user (end-user or researcher-user) more effective interaction with the system by seeing more information about the request sent to the system as well as, by better understanding of the results, how to refine the query iteratively.

Our qualitative evaluation of these multiple views in AIRS shows that each view has its own limitations and benefits. However, together, the views offer complementary information that helps user in improving his or her search effectiveness.

Keywords: Image database visualization, adaptive image retrieval system, multiple views interface, user-system interaction.

1. INTRODUCTION

Navigation and interaction are essential features in information visualization and image retrieval. It is essential for an image retrieval system to communicate information to users in the best possible way for the user to understand it. In this context, Croft[1] pointed out that interfaces for image retrieval systems should support features like query formulation, feedback, and presentation of retrieved information.

There are many proposed interfaces for introducing the query: annotation based, selection of features and/or objects from provided lists, sketching the image, selecting typical images of interest called query by example (see Ref. 2 for details).

One way to present the results to the user is by visualizing the projections of images into a 2D or 3D space, where similar images are positioned closer to each other. Some systems proposed to organize images into hierarchies (e.g. self-organizing maps[3] used in PicSOM). In these systems navigation through the data is restrictive because images are positioned at certain fixed distance in tree-like grids.

Pecenovic[4] proposed a visualization approach based on the idea that dynamic and interactive visualizations of data combined with search by queries will help the user to retrieve the desired information. Rubner[5] has proposed to use a low-level content based similarity metric to create visualizations of sets of images. Thumbnails of the images are placed by using the multidimensional scaling (MDS) method[6] on the display such that the distances on the screen reflect the real distances between the images as much as possible. Experiments[4] show that users preferred the results displayed into 2D maps over the ones displayed in lists, with faster results achieved in the 2D maps layouts.[5]

Further author information: (Send correspondence to A.D.M.)
E-mail: {axd9917,raghavan,skk0487,cice}@cacs.louisiana.edu, Telephone: 337-482-1231

Adaptive learning techniques are successfully applied to image retrieval systems. An AIRS* is based on the *relevance feedback* method. By using a relevance feedback approach, the system tries to reduce the existing gap between the high-level semantic concepts existing in user's mind and expressed as queries, and the low-level features describing the images. Many image retrieval systems use a *query by example* approach, in which the query consists of a set of images (represented by color histograms) from the image database.

Usually, the user-system interaction is built through an interface that allows the relevance feedback process to take place. Most image retrieval systems simply display the result list of images (or their thumbnails) to the user in a 2D grid.[8] This view displays all images or their thumbnails, without any inside information about the relationships between images. This is referred to in this paper as the list view.

An AIRS contains different types of information about an image database such as images, their similarities, and their feature representations. We identify two types of users for an AIRS: a user who seeks images whom we refer to as an *end-user*, and a user who designs and researches the database and configures the retrieval system parameters whom we refer to as a *researcher-user.* The end-user searches for images as pictures, whereas a researcher needs more detailed information about the images (such their similarities, their feature representations) and the system.

As a result, there is a need for different views that display the information at different levels of detail and can be selected by different users according to their needs. The end-user who searches for a better query uses a list view or a graph view. Then, the designer or the researcher-user, is a user that works intimately with the retrieval system or with the image database, and needs an interface that provides detailed and accurate content information (illustrated in the histogram views) about the database and the retrieval process.

Our interface includes three types of views: list views, graph views, and histogram views, where each view type presents a certain type of information. Whereas the list and histogram views display retrieval output in a traditional way, one displaying images and the other displaying feature (here, color) representations of these images, the graph views display structural representations of these images.

Reducing the gap between the user's high-level concepts (queries) and the system's low-level knowledge helps the user perform a better search. In our interface for the AIRS, we represent the similarity information between images from the result list at each feedback step by using visual attributes (colors, shapes, proximities). In this way, we display at once both types of information: the results returned by the system and the search specifications given by the user as query.

By bringing up the information from all previous steps, the interface helps user to decide the next step in query building. Based on the new and old information the user is able to decide how to change the query in order to satisfy his or her need. Therefore, the interface can be used as a *support tool for query formulation.* Each view has its own limitations and benefits. However, together, the views offer complementary information that helps user in improving his or her search effectiveness.

This paper is organized as follows. In Section 2, we present related work and our motivation, as well as the layout algorithms used for our views. Section 3 presents our proposed multiple views interface and its usage by the different types of users. In Section 4, we compare the multiple views. Finally, Section 5 concludes the paper and suggests future work.

2. RELATED WORK

Enhancing the visualization of the query result is a valuable way of helping the user to satisfy information needs. Therefore, there is a *need for general purpose user interfaces* for visual information retrieval that should include features like query refinement from examples. The interfaces should also include similarity display of the visual examples and of the results, and easy handling of the visual content. That is, there is a *need for query refinement interfaces*, which should allow iterative query refinement by relevance feedback from user in a perceptive way.

The aim is also to reflect only as much detail as necessary for the user to get the appropriate content information that will help throughout the searching process. Content information can help understanding new

*Our AIRS was introduced elsewhere.[7]

(or not seen) images by inferring the similarities between these images and the old (known, seen) images. By navigating through the visual information, the user can improve her or his mental image about the existing relationships between the data reflected in the query, which, in turn, will help the retrieval system to return results that better satisfy the user's request. In this context, a major challenge is *how to visualize the relationships between images* in order to make the contents of the digital libraries more accessible and manageable to users.

Another challenge in Image Retrieval is the *user subjectivity.* It is known that there are significant differences between rankings produced for different users or even between two runs by the same user at different times for the same initial query. Next, we present briefly the layouts used in our interface.

Inspired by the work done by Pecenovic and Rubner[4, 5] we build a multiple-view interface (Fig. 2) for our AIRS that provides three types of views: list view, graph views, and histogram views, where each view type presents a different type of information. Whereas the list and histogram views display retrieval output in a traditional way, one displaying images and the other displaying feature (color) representations of these images, the graph views display structural representations of these images. Besides the views used by Pecenovic and Rubner we add the histogram views to our interface because we identify two types of users for our system: end-user and researcher-user. Each type of user has different needs which the system tries to satisfy by providing more detailed information about the database. We study these views to see how much we can address and improve these issues. To build the views for our interface we used three different algorithms, which we briefly describe next.

Kamada-Kawai (KK).[9] Based on given distances between any two images from the database, the algorithm builds a general undirected weighted graph, where the length of an edge equals to the Euclidean distance between the two points, which in our system represent two images.

Force-Directed (FD).[10] This algorithm builds an undirected graph with uniform edge lengths (no weights) for a multidimensional vector dataset. The method distributes the points evenly in the frame, with a nice spread layout.

Parallel Coordinates (PC).[11] The algorithm provides a very simple representation of high dimensional vectors on a plane. A parallel coordinates visualization assigns one vertical axis to each feature (color in our case). All axes are parallel to each other, and equally spaced horizontally. Each feature is plotted on its own axis by using its frequency value, and the points from the adjacent axes are connected by straight lines. Thus, a point in a n- dimensional space becomes a polygonal line with $n-1$ lines connecting the n color frequencies. The order in which the axes are drawn is arbitrary. Thus, different orders can produce different representations.

Herman[12] pointed out some existing visualization problems, encountered especially when dealing with graphs, such as planarity, predictability, time complexity, density, overlapping, the size of the image list to be displayed. We do not address these layout problems here, but we simply identify which of them exist in our views. Next, we try to understand how this interface helps the user in building a better query, and which differences exist between the multiple views. Next section presents our proposed multiple views interface.

3. MULTIPLE VIEWS INTERFACE FOR AIRS

3.1. Interface for Different Types of Users

Adaptive learning techniques are successfully applied to information retrieval systems. An AIRS is based on the *relevance feedback* method. By using a relevance feedback approach, the system tries to reduce the existing gap between the high-level semantic concepts existing in user's mind and expressed as queries, and the low-level features describing the images. Many image retrieval systems use a *query by example* approach, in which the query consists of a set of images (represented by color histograms) from the image database.

Usually, the user-system interaction is built through an interface that allows the relevance feedback process to take place. For a better user-system interaction, we add a dynamically clickable interface component to our AIRS. By displaying both images and different statistics for these images, our interface aims to be a better tool for a user to give feedback.

An AIRS is mainly used by two types of users, a user who seeks images whom we refer to as an *end-user*, and a user who designs and researches the database and the retrieval system whom we refer to as a *researcher-user.* As

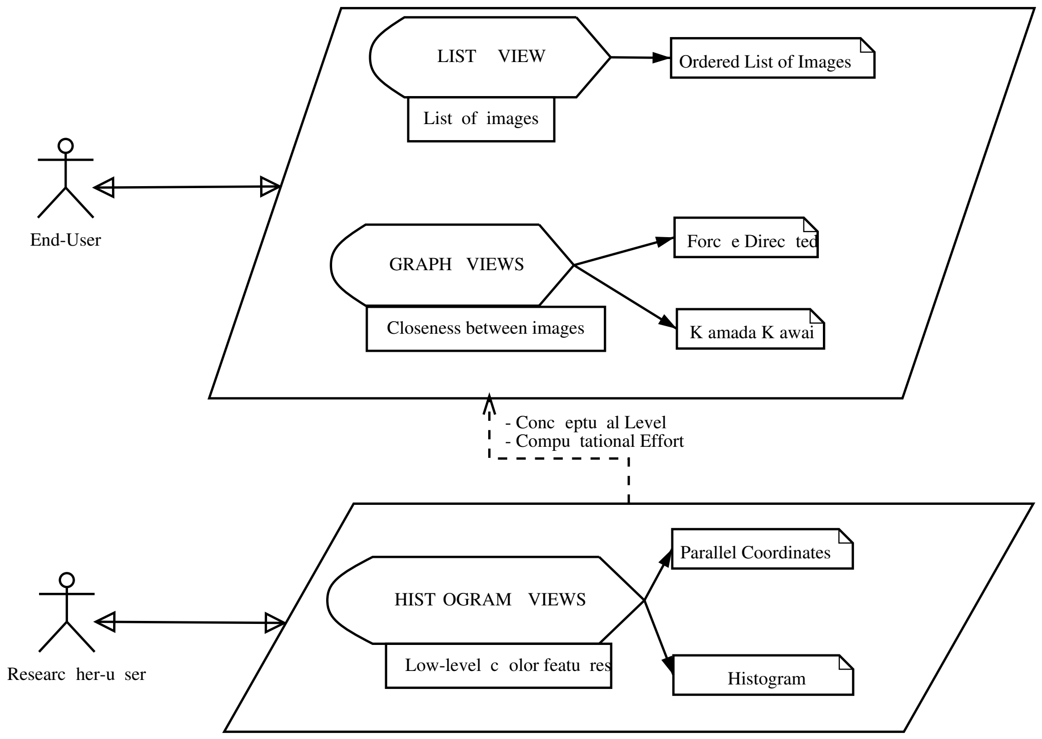


Figure 1. Conceptual views. The figure illustrates the conceptual levels of our views and their corresponding types of users. Each view type has attached a brief description of its layout, and the kind of algorithms used to build it.

both types of users seek for image information, their levels of information need differ. For example, the end-user searches for images as pictures, whereas a researcher needs detailed information about the representations of the images and the system.

As a result, there is a need for additional views that display the information at different levels of detail and can be selected by the users according to their needs. For this, we make a direct correspondence between the types of information needed by the different types of users and the visual information that is displayed for them.

First, the end-user who searches for a better query can use a list view or a graph view. Whereas the former ignores to display any existing relations between images, the latter is based on these relations. Actually, experiments[4, 5] show that users prefer the results displayed into graphs over the ones displayed in list views. Then, the designer or the researcher-user is a user that works effectively with the retrieval system or with the image database. Therefore, an interface that provides detailed and accurate content information (such as histogram views) about the database and the retrieval process will facilitate this user's search. Note that the researcher will try to see these images through the computer's "eyes".

Figure 1 illustrates the conceptual levels of our views and their corresponding types of users. Our many views are like users with different needs, each type of need being satisfied by a particular view. The interface acts as *a researcher's tool* or *an end-user's tool* depending on the user. The interface allows users to switch between the multiple views at each step, the choice depends on how much effort a user wants to put into this search process.

The multiple views are organized hierarchically according to the level of information that they display. For example, if one makes an analogy with a document organization, then at the top of the hierarchy is the list display, which acts like an general overview (or contents) of the document. The graph views follow in the hierarchy, and act like an abstract by displaying the relationships between the images from the result list. The histogram views, which display detailed information about the image representations, are at the bottom of the hierarchy and act like the content of the document.

However, if we look at these layouts from another point of view, both list and histogram views are detailed and accurate layouts, because they display the information as is without considering any relationships between the images. On the other hand, graph displays are abstract (structured) and approximate layouts since they symbolize images and their relationships. Note that graph views display more information than the list views and less than the histogram views. Therefore, maybe they can act as a bridge between the two conceptual levels and help user provide more clear requests, which in turn, will help the retrieval system to better satisfy user's needs. In this case, the graph views could address the needs of both users, end-user and research-user.

3.2. Multiple Views Interface for AIRS

Many image retrieval systems display the result list of images to the user by using a list view. However, in our system, the result list of images is presented to the user in a multiple views interface.

Each image is mapped to a node on the visualization map. The relationships between images are reflected by the spatial arrangements and by the colors of these nodes on the map. Any node in the visualization can be selected (click on the image point), causing the image that it refers to, to be returned in a window, in which the user can enter his or her feedback (relevant or non-relevant). As the user presses "Submit", the system returns a new ranking of the whole image database, to which we refer to as the result list. Then, the system splits the new result list into several sublists and displays each sublist on a separate page display. The user must introduce the desired number of images to be shown per page display. Figure 2 illustrates a page display of ten images. The "Next" and "Previous" buttons for each display allow user to navigate through the results list page by page (Fig. 2).

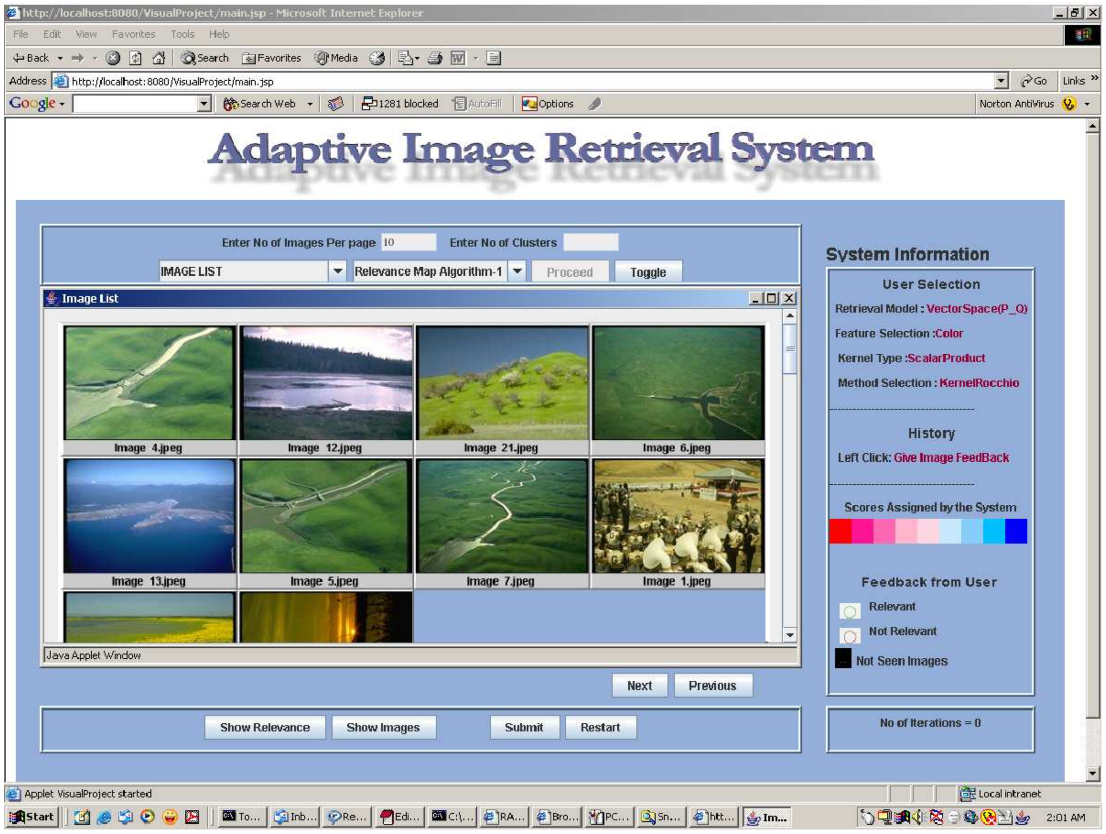
3.2.1. Multiple views description

Our interface includes four views: a standard list view (Fig. 2), two graphs views (Fig. 3 and 4), and two histogram views (Fig. 5 and 6). The user can switch at any moment between these different displays by pressing the "Toggle" button.

- *List view.* The list view is currently used by most of the image retrieval systems. This view displays all images or their thumbnails, without any inside information about the relationships between images.
- *Graph views.* The graph views denote the images as small rectangles on a 2D plane, based on distances[†] between images, with the closest two images connected through an edge by using KK or FD algorithm. Each such rectangle has dynamic coordinates (that change from an iteration to the next iteration) and an action associated with it. When the mouse passes over the image rectangle, a left click on the colored rectangle pops-up a window for user feedback.
- *Histogram views.* The histogram views, where we include the histogram (Fig. 5) and the parallel coordinates (Fig. 6) views, present the image at a low-level, at the feature representation (colors) level. By relying on the features representing the images, these layouts display very detailed and accurate views of the relationships between images.

The information visualized through these views is presented in the next section.

[†]The distances are calculated by using the standard Euclidian distance formula.



System Information

User Selection

Retrieval Model : VectorSpace(P_Q)

Feature Selection :Color

Kernel Type :ScalarProduct

Method Selection : KernelRocchio

History

Left Click: Give Image FeedBack

Scores Assigned by the System

Feedback from User

Relevant

Not Relevant

Not Seen Images

No of Iterations = 0

Enter No of Images Per page 10

Figure 2. List view. This image displays the interface of the AIRS. The "System Information" box on the right presents a history of the system. The result list is displayed by using the selected view type in the center area, which is surrounded by buttons for different inputs.

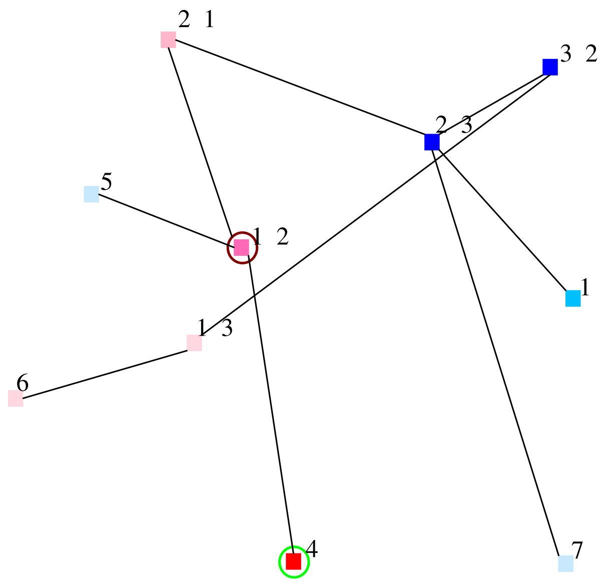


Figure 3. Force-Directed view.

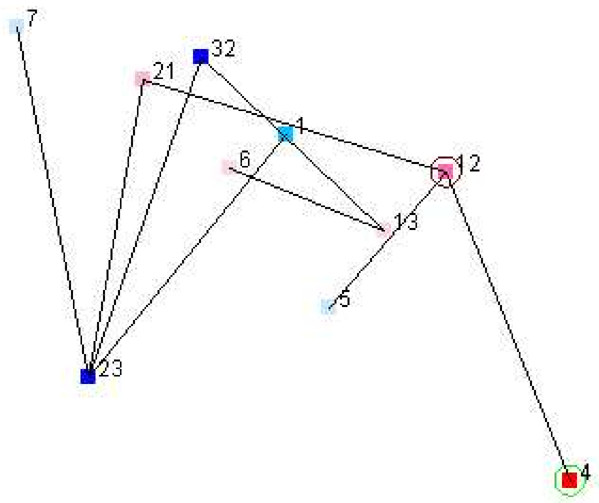


Figure 4. Kamada-Kawai view.

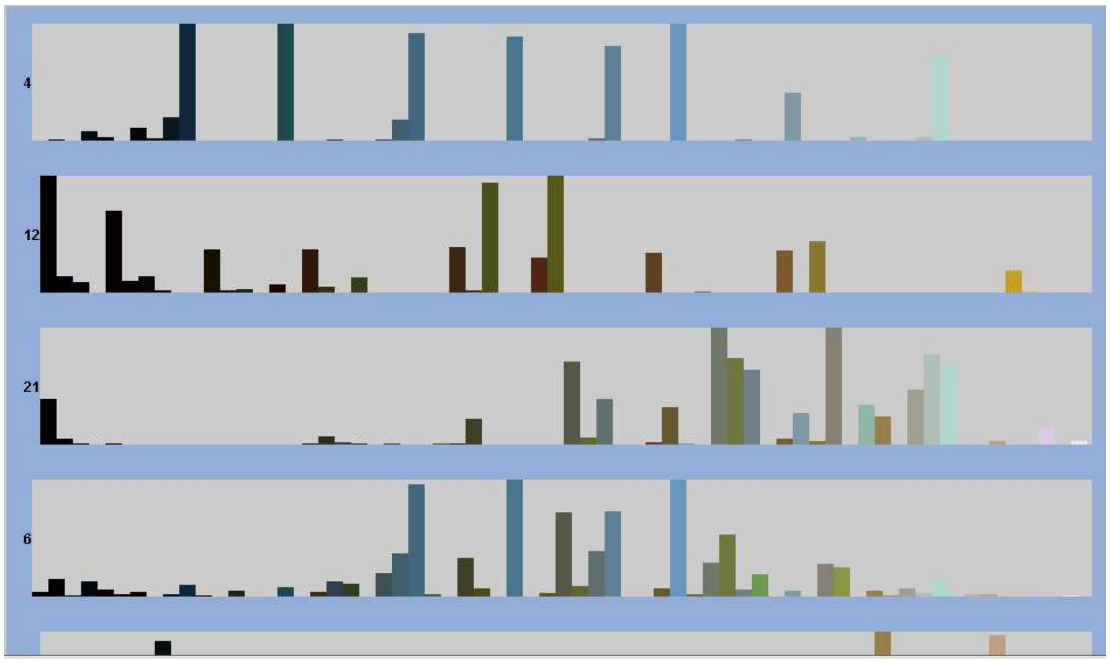


Figure 5. Histogram view.

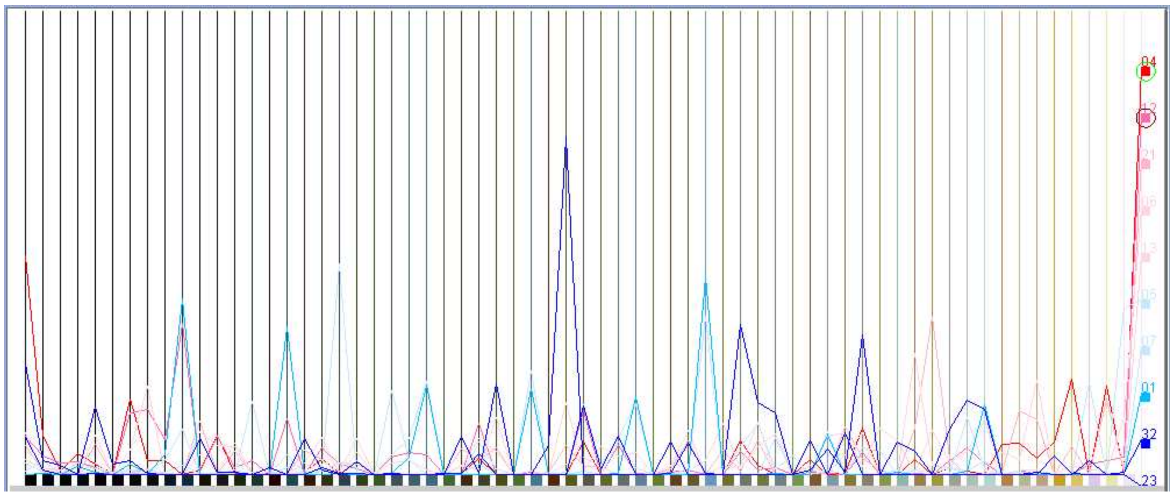


Figure 6. Parallel Coordinates view.

3.2.2. Visualising relations between images

A way to reduce the gap between the user's high-level concepts (queries) and the system's low-level knowledge is to help the user understand and perform better query search. For this, association of color, shape and proximity can be used to increase the amount of information to display. Since users perceive these visual attributes very easily, they can be used together to communicate more information to the user faster. Therefore, by associating these visual attributes we can display both relevance judgments, given by the system and by the user, of an image with respect to a query.

In our interface, we represent the similarity information between images from the result list at each feedback step by using color, shape, and position attributes. By positioning the images (their associated points or rectangles) in a meaningful way on the screen, the user can get a feeling of their closeness with respect to the given query. As a result, we get a spatial arrangement of images based on correlations between them (Fig. 3 and 4). In the following, we describe how we associate relevance assessments to visual elements in our interface.

Shape *symbolizes the relevance provider.* In AIRS, we deal with two types of relevances: one coming from the user as feedback, and the other one coming from the system as a result of the retrieval process.To differentiate between the two providers we use circles for the user, and rectangles for the system (for e.g. node 12 in Fig. 3 reflects both types of relevances).

Color *symbolizes the relevance attribute.* Our interface reflects the degree of relevance for each relevance provider separately, by using color, as follows:

- *"Temperature" like colormap for system relevance.* Our display is a 2D plane where small colored rectangles, that symbolize images, are filled with a color that symbolizes the image's relevance to the query, given by the system. For the first display, all images are "not relevant" and are displayed with the same color (i.e. black). Then, the color of the rectangle is dynamically determined by the system at each feedback step based on the image's rank in the results' list. Therefore, the colors of the rectangles act as a relevance color map returned by the system. Note that some images might share the same color (for e.g. nodes 5 and 7 in Fig. 3), which means that the retrieval system ranked them very close to each other with respect to the given query.

- *Green/brown colors for user relevance.* Initially, all images are in a random order as there is no feedback from the user (no circle). Once the user gives feedback on an image, his or her relevance is illustrated as a circle surrounding the image rectangle. A brown circle shows a non-relevant image, whereas a green one stands for a relevant image (for e.g. nodes 12, and 4 in Fig. 3, respectively). Therefore, we represent the user relevance of each image to a given request by using color.

***Position** (edges) symbolizes proximities between images.* The display distances are based on the similarities between images and not on the user's query. To display the images on a plane, algorithms for placement of points on a plane are used (Force-Directed and Kamada-Kawai algorithms). Both algorithms build a MST (minimum spanning tree) for a set of given points and displays them on the screen (see Fig. 3 and 4). Therefore, both layouts use edges to show the closeness between two images.

To summarize, we represent the relevance of each image to a given request by using its color and its arrangement on the plane. Further, we distinguish the relevances by using different shapes, such as a rectangle for the system relevance resulting from the learning process and a circle for user's preference. By doing this, the user can see the differences (if any) between the two (system's relevance versus his or her own relevance). In other words, the association of the visual attributes can suggest an image of the query. Next, we describe how to obtain the colors for the rectangles.

Building the colormap. We use a temperature map display to present to the user the system ranking of images according to his or her feedback. For this, we use 9 colors, ranging from red (as a most relevant image) to pastel-pink and white for less relevant images, and from pastel-blue for the non-relevant images with a higher rank to blue for the non-relevant images with lower rank (shown in the right panel of Fig. 2).

At each feedback step, the system reads all ranking (ordered) values returned by the system. The idea is to divide their range interval (the $(maxRank, minRank)$ interval) in sub-intervals and to assign the colors within these sub-intervals. There are two ways to assign the colors. The first method splits the ranking interval into pages, and then each page is considered to be a full colormap that is further divided into 9 equal sub-intervals. The second method divides the ranking interval into 9 equal sub-intervals to which the colors are assigned, and then each sub-interval is split into pages. Note that the only difference between the two algorithms lies in the number of images the system considers while assigning the colors. The user can chose either algorithm by selecting the corresponding name from the algorithm list (see Fig. 2). Next section discusses the role of the interface in the query formulation process.

3.3. Interface for Query Formulation

To build an AIRS is a difficult task by itself. Thus, if we can provide a *tool to help user understand the system problems and limitations* (and communicate them to the user), it will be like a next step in a further improvement of the system. It is known that the existing AIRSs are facing different problems, like the existing gap between concepts and representations, user subjectivity, etc. Therefore, an interface tool such as the one we propose can offer support in designing the system. This type of interface can help user understand the retrieval system, as well as the database with all existing relationships between the images. Therefore, it acts as a tool for image data exploration and analysis.

The interface can be used as a *help tool for query formulation.* By bringing up the information from all previous steps, the interface helps user to decide the next step in query building. Based on the new and old information the user is able to decide how to change the query in order to satisfy his or her need. At each retrieval step, the reorganization of consequent user interaction involves the whole database. Some images will disappear from the display, and some will appear. But, the views display the relevances (given by user and system) by using visual attributes (see Sect. 3.2.2) that help the user understand the relationships between the images.

In very general terms, the role of our interface is to focus the user attention on certain relations between images that, given the current meaning, are relevant. The system interface does this by displaying the relations between images according to the similarity criterion used to define the "meaning" or semantics of an image. The full meaning of an image depends not only on the image data, but also on the interpretation, i.e. the user perception of the image. Therefore, the query formulation should be seen as a process in which meaning is created through the interaction of the user and the images.

According to Santini[13] the meaning of an image is defined only in the context of a query, and can only be revealed in the context of the whole database. The meaning of an image is a set of relations between that image and the other images in the database. These relations are expressed as dissimilarity, or distance between two images and are depicted by the graph views in our interface.

On the other hand, the closeness between images (image semantics) is determined by the specific query that the user is asking. The meaning depends on the whole distribution of images in the database and on the metric used in the querying process. Therefore, the process of query formulation is a "conversational" activity during which the meaning of an image is created (see Ref. 13). By defining the meaning of an image as a result of the user-system interaction, the interface of the retrieval system becomes one of the most important components of the system.

While we want to display more useful information for the user to be able to better satisfy his or her needs, we also want to keep the display as simple as possible. The reason is that a simple visual interface is always easier and faster to read than a more complex one. For this, in next section, we study the views with their benefits and limitations, in a concrete case.

4. CASE STUDY - COMPARISON BETWEEN THE MULTIPLE VIEWS

4.1. Experimental set-up

To be able to compare our multiple view interface, we run the retrieval system for a given query. Our selected query is to find all images which contain only land surfaces covered with grass, i.e. images with only shades of green (images 04, 05, 06, and 07 in our database). Due to space limitations, we present only one result list obtained after several feedbacks from user, and depicted by all our views (Fig. 2, 3, 4, 5, and 6). We choose to present our different views after some feedback from user in order to include in our displays all types of information the interface can offer at one retrieval step necessary for better understanding and comparison of the views in our study.

In the following, we describe briefly the user-system interaction process in our example. At the beginning, the user selects the number of the images to be shown per page (10 in our example), the desired graphical view (here, we include all views for comparison purpose), and the color mapping algorithm (first, in our case). After the user sends this information to the system, the system responds with a display of 10 randomly chosen images. At this step, all images are "not seen" (i.e. color black and no surrounding circle). The user can navigate through the whole collection of images, page by page, by pressing the "Next" and "Previous" buttons.

However, in our example, the user gives a non-relevant attribute to 5 images (25, 26, 27, 28, and 29) from the set of 10 images displayed on the first page. Then, the system assigns colors to the images. According to the color map, the user can recognize the top most ranked image(s) returned by the system (color red, followed by shades of red until light pink) and can give more feedback (here, 39, 24, 16, 12, as non-relevant, and 4 as relevant). We present the results by using all views, after this feedback. Next, we discuss these views.

4.2. Discussion on the multiple views

All views reflect semantic information at different levels of abstraction through a visual layout. To help the user, this visual layout must be easily readable. The more crowded a display is, the more difficult it is for a user to distinguish between the points and to select the desired images for feedback.

In our example views, we noticed the following factors that influence the crowdedness of our layouts: overlapping colors, overlapping edges, crossing edges, number of images per page, and the number of features per page for the histogram views. Next, we compare our views along these factors.

Overlapping colors. Notice that in our example this factor does not appear in the histogram and list views, it can sometimes occur in the FD graph view, and it can happen in KK and PC views quite frequently. In the latter, we can have two or more images within the same relevance interval, i.e. with the same color (like images 05 and 07 in Fig. 6). In this case, it is difficult for the user to say which line (or rectangle) pertains to which of these images.

Overlapping edges. As we can see, there may be some overlapping edges in KK graph view and more often in PC view (Fig. 6) between different string lines (or images). In our example, we have overlapping edges of images with the same line color, like images 06 and 13 for some shades of light green, but also between images with different line colors, like images 01 and 13 for some shades of dark blue. In the PC view, the overlapping edges is a desired factor, but it makes the layout not readable if the number of images is too big.

Crossing edges. This factor is not an issue for the list display and the histogram display, but the PC view is based on it for comparing images. While both graph algorithms display the same MST, when comparing them, one can easily notice that KK displays edges without too much distortion, while FD does have a nice display without any edge weight consideration, but with more spread nodes and less crossing edges. Although KK algorithm reproduces the real closeness between images as the shortest path between them, this information might be difficult to read due to the number of the crossing edges.

Number of images. Another reason for a crowded display is the number of the displayed images. Whereas all views depend on this factor, their maximum limits differ. Out of all views the PC view requires the smallest number of images for a nice layout. For example, we have 10 images in Fig. 6, and the display is quite crowded already. This happens maybe due to the nature of the image histograms.

Number of features (colors, here). This factor is characteristic only to the histogram views, as the other views do not display images at the feature level. Both views display the same low level features (colors, 64 in our image representations). We notice that there is a trade-off between the user and the system desired number of features (colors). Whereas a small number is not enough for representing an image due to loosing the discrimination power, it will be easier for user to see its representation. Also, as colors are displayed quite small due to the window size limitation, a larger number of colors will result in an even more crowded display. This shows that there is a trade-off between the user visual limitation and the computer visual limitation.

Why multiple views interface for our AIRS? As we can see from the above discussion, each view presents problems and benefits. But the limitations posed by one view can be overcomed by another view, i.e. the views offer complementary information from which the user can benefit.

5. CONCLUSIONS

In this paper we have proposed a new multi-view interface for our AIRS, in which each view displays different information required by users according to their needs (end-user or researcher-user). This study provides a framework within which future experiments can be carried out to understand how well the views are deployed.

The users of the proposed interface have an active role in exploring the semantics of an image, which is a product of the user-system interaction mediated by the interface. Our interface illustrates different relations between images by using visual attributes (colors, shape, proximities). With such views, the user can see the relations between images, as well as better understand the results, and how to refine the query iteratively.

The interface allows user more interaction with the system by seeing more information about the request she or he sends to the system. The interface can be used as a help tool for query formulation. Thus, visual interfaces aim to support retrieval as well as browsing. Our qualitative evaluation of these multiple views in an adaptive image retrieval system shows that all views present limitations and benefits. However, the views offer complementary information that helps user in his or her search. Our work in progress deals with scalability issues posed by each view. Future work includes evaluation of the visual interface in the query formulation process.

ACKNOWLEDGMENTS

The authors would like to thank to Ryan Benton for useful discussions about building the interface.

REFERENCES

1. W. B. Croft, “What do people want from information retrieval?,” *D-Lib Magazine* , 1995.
2. R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*, Addison-Wesley, New York, 1999.
3. T. Kohonen, “Self-organization of very large document collections: State of the art,” in *Proceedings of the 8th International Conference on Artificial Neural Networks, ICANN'98*, L. Niklasson, M. Bodén, and T. Ziemke, eds., **1**, pp. 65–74, 1998.
4. Z. Pecenovic, M. N. Do, M. Vetterli, and P. Pu, “Integrated browsing and searching of large image collections,” in *Proceedings of the 4th Int. Conf. on Visual Information Systems (VISUAL)*, pp. 279–289, 2000.

5. Y. Rubner, C. Tomasi, and L. J. Guibas, “A metric for distributions with applications to image databases,” in *Proceedings of the 6th IEEE International Conference on Computer Vision*, pp. 59–66, 1998.
6. R. N. Shepard, “The analysis of proximities: Multidimensional scaling with an unknown distance function,” *Psychometrika* **27**, pp. 125–140, 1962.
7. A. Doloc-Mihu, V. V. Raghavan, and P. Bollmann-Sdorra, “Color retrieval in vector space model,” in *Proceedings of the 26th International ACM SIGIR Workshop on Mathematical/Formal Methods in Information Retrieval MF/IR*, ACM, ed., pp. 1–15, 2003.
8. R. C. Veltkamp and M. Tanase, “Content-based image retrieval systems: A survey,” Tech. Rep. UU-CS-2000-34, Ultrecht University, URL http://www.aa-lab.cs.uu.nl/cbirsurvey/cbir-survey/, 2000.
9. T. Kamada and S. Kawai, “An algorithm for drawing general undirected graphs,” *Information Processing Letters* **31**, pp. 7–15, 1989.
10. T. M. J. Fruchterman and E. M. Reingold, “Graph drawing by force-directed placement,” *Software - Practice and Experience* **21**, pp. 1129–1164, 1991.
11. A. Inselberg, “Multidimensional detective,” in *IEEE Symposium of Information Visualization*, I. C. Society, ed., pp. 100–107, 1997.
12. I. Herman, G. Melancon, and M. S. Marshall, “Graph visualization and navigation in information visualization: a survey,” *IEEE Transactions on Visualization and Computer Graphics* **6**, pp. 1–21, 2000.
13. S. Santini, A. Gupta, and R. Jain, “A user interface for emergent semantics in image databases,” in *Proceedings of the 8th IFIP Working Conference on Database Semantics (DS-8)*, pp. 123–143, 1999.