Retrieving and Localizing Objects in Images on the WWW

Theo Gevers, Frank Aldershoff, Peter Vreman, Arnold W.M. Smeulders
Faculty of Mathematics & Computer Science, University of Amsterdam
Kruislaan 403, 1098 SJ Amsterdam, The Netherlands
E-mail: {gevers, aldersho, pfvreman, smeulders}@wins.uva.nl

Abstract

In this paper, we give an overview of the PicToSeek system for exploring visual information on the World Wide Web. PicToSeek automatically collects, indexes and catalogs visual information entirely on the basis of the pictorial content. PicToSeek allows for content-based image retrieval conducted in an interactive, iterative manner guided by the user by relevance feedback. Relevance feedback can be seen as a method of feature selection and weighting. The PicToSeek system has been implemented based on the client-server paradigm. The client is a Java Applet and takes care of interactive query formulation, the display of the results, and the relevance feedback specification given by the user. The server is a Servlet using C-libraries and takes care of the image feature extraction, feature weighting from relevance feedback, k-nearest neighbour feature classification, and image sorting. The system is available at http://www.wins.uva.nl/research/isis/zomax/.

1 Introduction

Image databases are becoming more pervasive due to the low cost of cameras, scanners and storage devices. Although today's technology enables efficient storage, search methods for retrieving pictorial entities from large electronic image archives are limited. Traditionally, verbal descriptions, such as keywords, file identifiers, or text, have been used to describe and retrieve images. However, it is difficult or impossible to fully capture real-world objects by words alone. Furthermore, adding verbal descriptions is time consuming, impractical or even impossible (e.g. Internet). Hence, in those cases, the capabilities of current text-based methods for retrieving images is limited. Consequently, there is a growing interest in *content-based* image retrieval. This approach circumvents the problem of using text as the basis for image retrieval. The search is carried out from a pictorial specification. Image retrieval by content is then the process to compute to what extent images in the database correspond to the pictorial specification.

In this paper, we give an overview of the image search engine, called PicToSeek, for searching images on the World Wide Web. In the first stage, PicToSeek collects images on the World Wide Web by means of autonomous Web-crawlers. Then, the collected images are automatically cataloged into various image styles and types: JFIF-GIF, grey-color, size, date of creation, and color depth. Further, the system automatically classifies (by supervised learning) images into the following classes: photograph-synthetic, (photographs) indoor-outdoor, (photographs) portraits, (synthetics) buttons. After cataloging images, invariant color image features are extracted from the images. Color invariant features are properly integrated to produce a high-dimensional image index independent of the accidental imaging conditions. When images are automatically collected, cataloged and indexed, PicToSeek allows for fast on-line image search by combining: (1) visual browsing through the precomputed image catalogue; (2) query by pictorial example; (3) query by image features. The content-based image retrieval process is conducted in an interactive, iterative manner guided by the user by relevance feedback. Relevance feedback can be seen as an method of feature selection and weighting. From the user feed-back giving negative/positive answers, the method can automatically learn which image features are more important. The effect of relevance feedback is to "move" the query in the direction of the relevant images and away from the non-relevant ones. In this way, the image retrieval system as a whole increases the probability of finding the right images one is looking for. Consequently, new visualization techniques are used to establish communications between system and user. These visualization techniques are intuitive and simple enabling non-trained persons to perform effective image retrieval.

This paper is organized as follows. In Section 2, related work is discussed. The image domain and pre-classification scheme is presented in Section 3. An overview of the PicToSeek system is given in Section

4. The implementation of the PicToSeek system on the World Wide Web is discussed in Section 5.

2 Related Work

Very large digital image archives have been created and used in a number of applications including archives of images of postal stamps, textile patterns, museum objects, trademarks and logos, and views from everyday life as it appears in home videos and consumer photography. Moreover, with the growth and popularity of the World Wide Web, a tremendous amount of visual information is made accessible publicly. As a consequence, there is a growing demand for search methods retrieving pictorial entities from large image archives. Currently, a large number of text-based search engines are available and they have been proven to be very successful in retrieving documents. To locate pictorial information, these text-based search engines assume that textual descriptions of the visual data are present. However, people are reluctant in verbally categorizing visual information. Moreover, using text as the basis for retrieval is almost always inadequate due to the semantic richness of pictorial information. Often no textual description of the pictorial information is present at all. Hence, in those cases, the capabilities of current text-based search engines for retrieving images is limited.

Attempts have been made to develop general purpose image retrieval systems based on multiple features (e.g. color, shape and texture) describing the image content [1], [2], [3], [11], [18], for example. Further, a number of systems are available for retrieving images from the World Wide Web, for example [13], [14], [15], [17]. These systems retrieve images on the basis of keywords and/or the image content.

However, these image retrieval methods are optimized for one specific application and hence problems occur when converting the retrieval scheme to another application domain. Also the image features used for search are usually dependent on the imaging conditions resulting in erroneous retrieval results. The practical usefulness of a retrieval system where only identical views can be retrieved will be an order of magnitude less than a system where objects can be searched for on the basis of random views. Further, the query formulation is often not easy to understand for users without any image processing background. In particular, most users find it difficult to formulate queries that are well designed for retrieval purposes: "How do I formulate a query to retrieve an image of a bear". Due to inadequate query formulation, the system will retrieve (too) many non-relevant images. Also image processing background is required when interpreting the retrieval results: "Why did the system retrieve a logo-image of red car when I searched for a logo-image containing a bear". To this end, the system should make clear the reason (how and why) the images have been retrieved and consequently to enable the user to adjust/improve the pictorial query in an intuitive and interactive manner. It is our point of view that content-based image retrieval needs user interaction to select relevant candidate images to refine the query. User friendly visualization tools should be provided indicating why and how the query formulation has matched the retrieved image. Further, the application dependency of the retrieval system should be reduced to a minimum.

3 Image Domain

On the Web, images are published by different people with various interests. As a consequence, a collection is created of images of various types and styles. Our aim is to crawl images efficiently by Web-robots and preclassify them into different styles and types. To that end, the Web-crawler is briefly outlined in Section 3.1 to acquire images from a variety of sites. In Section 3.2, the image classification method is briefly demonstrated.

3.1 Image Collection

A Web-crawler has been implemented. A one-sided view on the Web is prevented in two ways. First, the choice to use a general index page for the starting point considerably reduces the chance that only a specific part of the Web is explored. Secondly, the use of a breadth-first strategy reduces the chance that all pictures come from just a few sited. The Web-robot down loaded over 100.000 images of the GIF and JFIF (JPEG) formats.

The results of the Web-crawler show a broad spectrum of image addresses. The 100.000 images in the final test set originate from 1345 sites. The number of images per class format in the test collection is 20 % photographs and 80 % synthetic images. 11 % of the collection are grey value images.

3.2 Image Classification

We classify images according to the creation method of the image into: *Photographical images:* pictures of 2D and 3D objects in real-world cluttered scenes as recorded by a sensor (camera); *Synthetical images:* computer-generated pictures of abstractions of 2D and 3D real-world objects such as icons, graphics, drawings and legend baring images.

After classifying images into photographical and synthetical images, we further classify photographical images into portraits (the image contains a (substantial) face), indoors (images are taken from indoor scenes), outdoors (images are taken from outdoor scenes). Further, synthetical images are classified into button or non-button images.

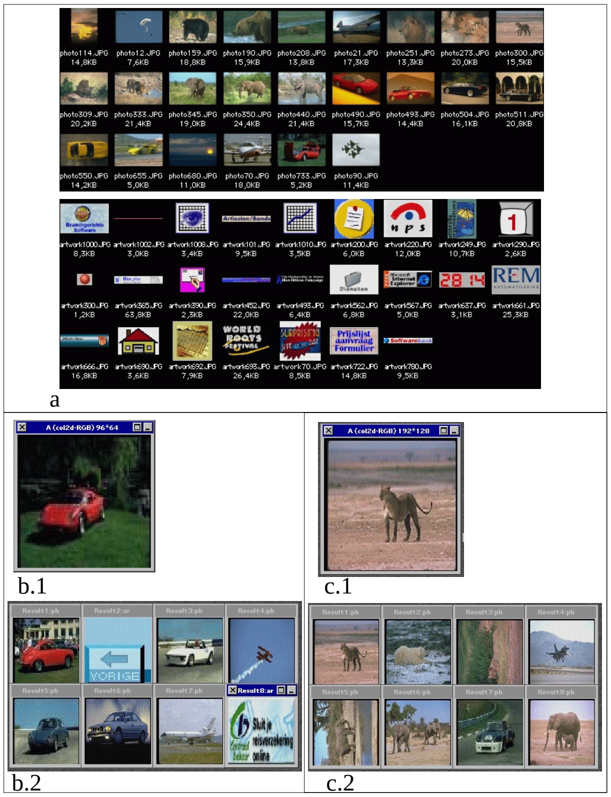


Figure 1: *a. The test sets of photographical and synthetical images. b.1-b.2 Typical photographical query and corresponding result. c.1-c.2 Typical photographical query and corresponding result.*

To illustrate the image classification scheme, we now briefly focus on classifying images into photographical/synthetical.

Photographical/synthetical classification: The following image features are computed from the color images: *Color variation*: The number of distinct hue values in an image relative to the total number of hues. Synthetic images tend to have fewer distinct colors then photographs. *Color saturation*: The accumulation of the saturation of colors in an image relative to the total number of pixels. Colors in synthetic images are likely to be more saturated. *Color transition strength*: The pronouncement of color invariant edges in an image. Synthetic images tend to have more abrupt color transitions than photograph images.

To classify images into photographical and synthetical images, we used the k-nearest neighbor classifier. We assessed the precision of the automated image type classification method, for a training and test set of 200 and 50 images respectively from each class. The test sets of photographical and synthetical images are shown in Fig. 1.a. Typical queries are shown in Figs. 1.b.1 and 1.c.1. The corresponding results, based on saturation and edge strength, are shown Figs. 1.b.2 and 1.c.2. Based on edge strength and saturation the 4-nearest neighbor classifier provided a classification success of 90% (i.e. 90% were correctly classified) for photographical images and 85% for synthetical images. From the results it is concluded that automated type classification provides satisfactory distinction.

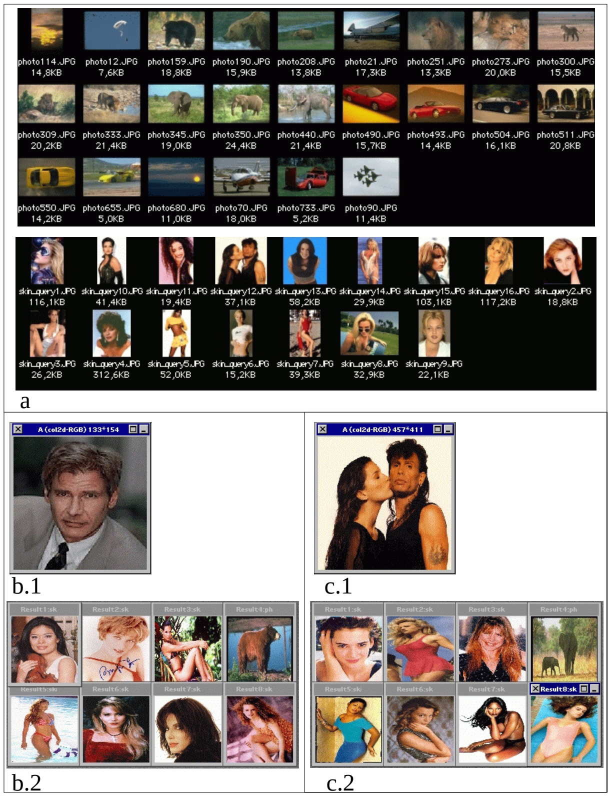


Figure 2: *a. The test sets of photographical and skin images. b.1-b.2 Typical skin query and corresponding result. c.1-c.2 Typical skin query and corresponding result.*

Skin-people classification: We have developed a skin-detector based on color invariant ratio's [7]. To classify images into people-portrait images, we again used the k-nearest neighbor classifier for a training and test set of 211 and 32 images respectively from each class. The test sets of images (not) containing people are shown in Fig. 2.a. Typical queries are shown in Figs. 2.b.1 and 2.c.1. The corresponding results, based on skin detector, are shown Figs. 2.b.2 and 2.c.2. Based on the skin detector the 4-nearest neighbor classifier provided a classification success of 81% for images containing skin.

4 System Overview

The basic idea to content-based image retrieval is to extract characteristic features from images in the database which are stored and indexed. This is done off-line. These features are typically derived from shape, texture or color information. The on-line image retrieval process consists of a query example image from which image features are extracted and matched with those derived from target images in the database.

We aim at a content-based image retrieval by relevance feedback. Relevance feedback is an automatic process designed to produce improved query formulations following an initial retrieval operation. The effect of such query alteration process is to "move" the query in the direction of the relevant images and away from the non-relevant ones. From the pattern recognition literature it is well known that k-nearest neighbor classifier will converge to the correct classification given sufficient examples. The advantages of relevance feedback are: 1. It shields the user from the details of the query formulation 2. It breaks down the retrieval process into a sequence of steps, designed to approach the wanted images gradually. 3. It provides a controlled query alteration process by emphasizing/deemphasizing features weights.

The major components of the PicToSeek system are described in detail below:

Interactive Query formulation

An image is sketched, recorded or selected from a repository. This is the query definition with the aim to find a similar image in the database. Note that "similar image" may imply a partially identical image (as in the case of finding stamps), or a partially identical object in the image (as in the case of a stolen goods database), or a similar styled image (as in the case of a fashion design support system).

PicToSeek offers snakes for interactive image segmentation, described in [5], for the purpose of content-based image retrieval by query-by-example. We proposed the use of color *invariant* gradient information to guide the deformation process to obtain snake boundaries which correspond to material boundaries in images discounting the disturbing influences of surface orientation, illumination, shadows and highlights. The key idea is to allow the user to specify in an interactive way salient sub-images of objects on which the image object search will be based. In this way, confounding and misleading image information is discarded. In conclusion, PicToSeek offers interactive query formulation either by query (sub)image(s) or by offering a pattern of feature values and weights.

Image features

The essence of the query image is captured in a set of features suitable for the interpretation of the similarity. These features may cover each aspect of the image data, a measurement of intensity, shape, color or texture, movement or model adherence. The PicToSeek system concentrates on color features measured at salient shape positions in the images, enabling the recognition of real world objects from just a few data points (and their color values) in the image. These invariant features are properly integrated to produce a n-dimensional image index independent of the accidental imaging conditions.

Recently, in [4], [6], we have proposed and evaluated different color features for the purpose of image retrieval by histogram matching. Feature spaces were constructed on the basis of different color features representing the distribution of discrete color feature values in a n-dimensional color feature space, where $n = 3$ for RGB and rgb, and $n = 1$ for I, S and H. On the basis of the reported theory and experimental results given in [4], [6], it can be concluded that RGB and I are sensitive to shadows, surface orientation and illumination intensity. RGB, I, rgb and S are all sensitive to highlights. The color model m is insensitive for smoothly sloping highlights. H is independent of surface specularities. m is the only color model invariant to illumination color.

PicToSeek allows the user to choose the desired classes of invariance. For each image retrieval query a proper definition of the desired invariance is in order. Does the applicant wish search for the object in rotation and scale invariance? illumination invariance? viewpoint invariance? occlusion invariance? In the current state of the art of query engines, invariance receives little attention. But for large databases, the availability at the time of query definition is essential. In the PicToSeek system both viewpoint invariant color features, as well as illumination invariant features are included.

Feature Representation and Weighting

The image feature sets are represented by n-dimensional feature space. In this way, the domain dependent part of the whole image retrieval system is reduced to a minimum.

To be precise, let an image I be represented by its *image feature vectors* of the form $I = (f_0, w_{I0}; f_1, w_{I1}; ...; f_t, w_{It})$ and a typical query Q by

$Q = (f_0, w_{Q0}; f_1, w_{Q1}; ...,; f_t, w_{Qt})$, where w_{Ik} (or w_{Qk}) represent the weight of image feature f_k in image I (or query Q), and t image features are used for image object search. The weights are assumed to be between 0 and 1.

Weights can be assigned corresponding to the feature frequency ff as defined by:

$$w_i = \mathrm{ff}_i \tag{1}$$

giving the well-known histogram form where ff_i (feature frequency) is the frequency of occurrences of the image feature values i in the image or query. However, for accurate image object search, it is desirable to assign weights in accordance to the importance of the image features. To that end, the image feature weights used for both images and queries are computed as the product of the features frequency multiplied by the inverse collection frequency factor, defined by [12]:

$$w_i = (0.5 + \frac{0.5\mathrm{ff}_i}{\max\{\mathrm{ff}\}_{i=1}^t}) \log(\frac{N}{n}) \tag{2}$$

where N is the number of images in the database and n denotes the number of images to which a feature value is assigned. In this way, features are emphasized having high feature frequencies but low overall collection frequencies.

Matching measure

The actual matching process is to search for the k elements in the stored image set closest to the query image. As both the query images as the data set is captured in feature values, the similarity function operates between the weighted feature sets. Again, the make the query useful, attention has to be paid to the selection of the similarity function. A proper similarity function should be robust to object fragmentation, occlusion and clutter by the presence of other objects in the view.

In [4], [6], various similarity functions were presented and evaluated for image retrieval. From the results it can be concluded that retrieval accuracy of similarity functions depend on the presence of object clutter in the scene. The cross correlation similarity measure provides best retrieval accuracy without any object clutter. Given the weighted vector representation, the normalized cross correlation, defining the query-image similarity measure, is given by:

$$S(Q, I) = \frac{\sum_{k=1}^t w_{Qk} * w_{Ik}}{\sum_{k=1}^t (w_{Qk})^2} \tag{3}$$

The normalized cross correlation has a maximum of unity that occurs if and only if Q exactly matches I. Accurate image retrieval when using this similarity function due to the fact that this similarity function is symmetric and can be interpreted as the number of pixels with the same values in the query image which can be found present in the retrieved image and vice versa. This is a desirable property when one object per image is recorded without any object clutter. In the presence of object clutter and occlusion, highest image retrieval accuracy is provided by the following similarity function (e.g. histogram intersection) (L_1-norm);

$$S(Q, I) = \frac{\sum_{k=1}^t \min\{w_{Qk}, w_{Ik}\}}{\sum_{k=1}^t w_{Qk}} \tag{4}$$

Robustness against object clutter and occlusion is due to the fact that this similarity measure counts the number of similar hits and hence is insensitive to false positives.

Searching

In the field of pattern recognition, several methods have been proposed that improve classification automatically through experience such as artificial neural networks, decision tree learning, Bayesian learning and k-nearest neighbor classifiers. Except for the k-nearest neighbor classifier, the other methods construct a general, explicit description of the target function when training examples are provided. In contrast, k-nearest neighbor classification consist of finding relationship to the previously stored images each time a new query image is given. When a new query is given by the user, a set of similar related images is retrieved from the image database and used to classify the new query image. The advantage of k-nearest neighbor classification is that the technique construct a local approximation to the target function that applies in the neighborhood of the new image query images, and never construct an approximation designed to perform well over the entire instance space. To that end, PicToSeek uses the k-nearest neighbour classifier for image search.

Indexing

Because the k-nearest neighbor algorithm delays classification until a new query is received, significant computation can be required to process each new query. Various methods have been developed for indexing the stored images so that the nearest neighbors can be identified efficiently at some additional costs in memory, such as a k-d trees or R*-trees, [9] for example. Unfortunately, the complexity of these search algorithms grows exponentially with the dimension of the vector space making them impractical for dimensionality above 15. To overcome this problem, we use the SR-tree for image indexing and high-dimensional nearest neighbor search. It has been shown that the SR-tree outperforms the R*-tree [10].

Visualization

Visualization of the feature matching results is very important. In this way, the system gives the user insight in

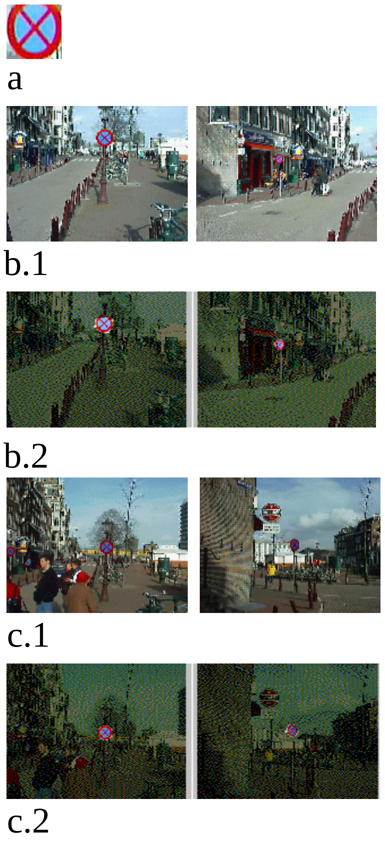


Figure 3: *a. Query image: traffic sign. b.1 Images in the database. b.2 Computation of the localization of the sign. c.1 Images in the database. c.2 Computation of the localization of the sign.*

the importance of the different features (even for those not used by the user during the previous search).

To achieve this, the back projection algorithm can be used [16]. However, this method is shape and scale dependent as the neighborhood-size and shape of the region to be considered should be known in advance. To that end, we proposed the use of a split-and-merge technique [8]. The procedure begins with an arbitrary partition of the image into equally-sized square regions. Regions will then be split if they don't satisfy a certain homogeneity criterion and four adjacent will be merged if their resulting region satisfies a certain homogeneity criterion. The homogeneity criterion is defined by the histogram intersection of the regions with the query example. This process continues until no more regions can be split and no more resulting regions can be merged. Finally, adjacent regions will be merged yielding the segmentation result. To illustrate the localization technique, consider the problem of retrieving and localizing traffic signs, see Fig. 3, in outdoor street-scenes. It is shown that the split-and-merge technique provides localization which is scale- and shape-independent. After localization, we use windowing and information display techniques to establish communications between system and user. In particular, ranked list of retrieved images can be graphically displayed for the user, and screen pointers can be used to designate certain images as relevant to the user's needs. These relevance indications are then further used by the system to construct modified feedback queries.

Relevance feedback

Relevance feedback is an automatic process designed to produce improved query formulations following an initial retrieval operation. Relevance feedback is needed for image retrieval where the users find it difficult to formulate pictorial queries which are well designed for accurate retrieval purposes. For example, without any specific query image example, the user might find it difficult to formulate a query (e.g. to retrieve an image of a car) by an image sketch or by offering a pattern of feature values and weights. This suggests that the first search operation should be conducted with a tentative, initial query formulation, and should be processed as a trial search only, with the aim to retrieve only a few useful images from the large image collection. These initially retrieved images should then be examined for relevance, and a (new) improved query formulation should be constructed with the purpose to retrieve more relevant images in subsequent search operations. Hence, from the user feed-back giving negative/positive answers, the method can automatically learn which image features are more important. The system use the feature weighting given by the user to find the images in the image database which are most similar with respect to the feature weighting. The feedback process can be represented graphically as a migration of the query vector from one area to another in the n-dimensional space.

To be precise, consider the weighted vector representation. The PicToSeek relevance feedback is formulated as follows [12]:

$$Q_1 = \alpha Q_0 + \beta \sum_{\text{rel}} \frac{D_i}{|D_i|} - \gamma \sum_{\text{nonrel}} \frac{D_i}{|D_i|} \tag{5}$$

where Q_0 and Q_1 denote the image feature vectors constructed for initial and the first iteration image example queries. The summation is taken over the relevant and non-relevant (sub)images. α, β and γ are appropriate weights. Notice that the original image query Q_0 is preserved in the feedback formulation.

In conclusion, the aim of relevance feedback is to produce improved initial query specification. In fact, the query modification moves the query into the desired direction. The above described relevance feedback process is simple and effective.

5 Implementation

The PicToSeek system is based on a client-server paradigm. The client part is a Java Applet and correspond to the graphical user interface. The client part takes care of interactive query formulation, the display of the results, and the relevance feedback specification given by the user. The server part of PicToSeek takes care of the image feature extraction, feature weighting from relevance feedback, k-nearest neighbour feature classification, and image sorting. The server is implemented in C. The interface between client (Java) and server (C) is written in Java. The Web-crawler, image analysis and feature extraction methods have been implemented in C. The server runs on a Ultra 10 Sun station with 300 Mhz.

The client and server components are described more in detail below:

Client site

Using a standard web-browser, the PicToSeek Applet is sent to the client. After the Applet has started, the user can load any image available at the WWW by giving the url-address. After the user has loaded an image, the user is allowed to specify (sub)images by the interactive snake segmentation method. After interactive query formulation, the user specifies the preferred invariance, and the similarity measure. Then, the image query formulation is send to the server. In conclusion, the client-part is a Java Applet and can be started by a standard web browser. The Java Applet allows the user to (1) select/load an (external) image, (2) select appropriate subimages of objects (instead of the entire image) on which the image object search will be conducted (3) select color features (invariants) and similarity measure (4) send the query formulation to the server.

Server site

The server receives the query image formulation send by the client. After receiving the query image, the server convert the image to the desired format, enabling the image processing routines, implemented in C, to extract the required image features. Query image features are weighted. In this way, features are emphasized having high feature frequencies but low overall collection frequencies. K-nearest neighbors are found in this weighted vector representation using the SR-Tree algorithm. The k-nearest neighbors are sorted with respect to their similarity and send back to the client for display. In conclusion, the server receives the image query formulation from the client. Then, the following operations are performed (1) image feature extraction (2) image feature weighting (3) the k-nearest neighbors are found and sorted (4) the results are send back to the client for display.

Examples

All queries follow the same scenario:

Step 0 **Image Domain Selection:** Visual browsing through the precomputed image catalogue;

Step 1 **Image selection:** select an image from the catalogue or capture the query image from an object by giving an URL-address.

Step 2 **Query image:** the query image is defined as an user-specified interesting part of the selected image.

Step 3 **Invariance selection:** the required invariance is selected from the list of available invariant indices.

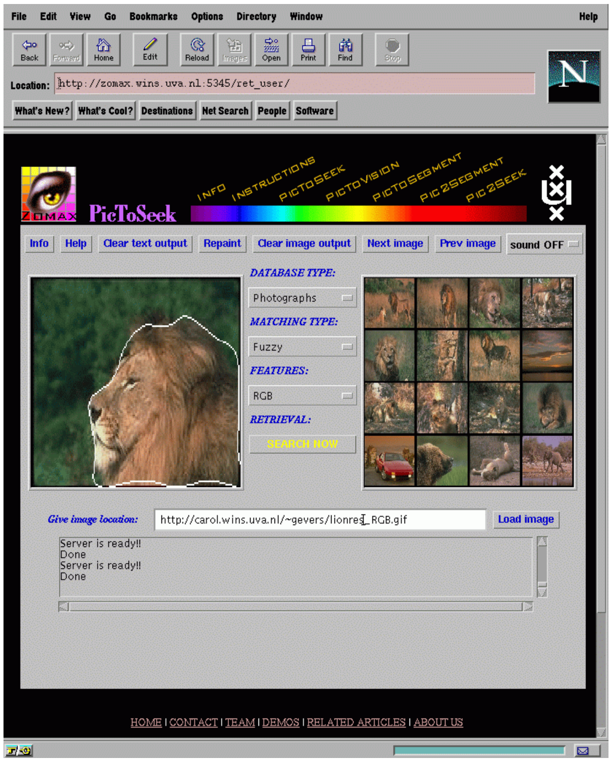
Step 4 **Search:** the same invariant indices are computed from the query and matched with those stored in the database hash tables.

Step 5 **Display:** an ordered list of most similar images is shown.

Step 6 **Image selection:** if the right image is found, the image can be displayed at full resolution.

Step 7 **Rerun:** if the right image is not found the query image is adjusted (go to step 1) or similar images are indicated to adjust/refine query definition (go to step 3).

To illustrate the query capability of the system, the typical application is considered of retrieving images containing an instance of a given object. To that end, the query is specified by an example image taken from the object at hand. A typical search operation is shown in Figures 4. The images come come from Corel © Stock Photo Libraries. The user has specified the region showing a lion. The region is used as the query. Images in the image database are compared to the lion query based on the color-texture information. After image matching, images are shown to the user in order of resemblance. Note that within the first 16 images, at least 12 images contain a lion.



Location: http://zomax.wins.uva.nl:5345/ret_user/

sound OFF

DATABASE TYPE: Photographs

MATCHING TYPE: Fuzzy

FEATURES: RGB

RETRIEVAL:

Give image location: http://carol.wins.uva.nl/~gevers/lionres_RGB.gif

```
Server is ready!!
Done
Server is ready!!
Done
```

Figure 4: *Content-based image retrieval by query-by-example based on the region denoting the lion (without the background) as specified by the user.*

6 Summary

In this paper, an overview is given of the PicToSeek system for exploring visual information on the World Wide Web. PicToSeek allows for content-based image retrieval conducted in an interactive, iterative manner guided by the user by relevance feedback. The PicToSeek system has been implemented on Internet by using the client-server paradigm. The client is a Java Applet and takes care of interactive query formulation, the display of the results, and the relevance feedback specification given by the user. The server is a Servlet using C-libraries and takes care of the image feature extraction, feature weighting from relevance feedback, k-nearest neighbour feature classification, and image sorting. The system is available at http://www.wins.uva.nl/research/isis/zomax/.

References

[1] Proceedings of First International Workshop on Image Databases and Multi Media Search, IDB-MMS '96, Amsterdam, The Netherlands, 1996.

[2] Proceedings of IEEE Workshop on Content-based Access and Video Libraries, CVPR, 1997.

[3] Flickner, M. et al, *Query by Image and Video Content: the QBIC system*, IEEE Computer, Vol. 28, No. 9, 1995.

[4] Gevers, T., *Color Image Invariant Segmentation and Retrieval*, PhD Thesis, ISBN 90-74795-51-X, University of Amsterdam, The Netherlands, 1996.

[5] Gevers, T and Smeulders W.M., *Interactive Query Formulation for Object Search* Visual99, Amsterdam, 1999.

[6] Gevers, T and Smeulders W.M., *Color Based Object Recognition*, Pattern Recognition, 32, pp. 453-464, 1999.

[7] Gevers, T and Aldershoff, F., *Skin detector for image classification* Technical report, University of Amsterdam, 1999.

[8] Gevers, T and Vreman, P., *Localization of Objects in Images* Technical report, University of Amsterdam, 1999.

[9] A. Guttman, R-trees: A Dynamic Index Structure for Spatial Searching, ACM SIGMOD, pp. 47-57, 1984.

[10] Katayama, N. and Satoh, S. *SR-tree: an Index Structure for High-dimensional nearest neighbor queries*, ACM SIGMOD, Arizona, 1997.

[11] Pentland, A., Picard, R. W. and Sclaroff, S., *Photobook: Tools for Content-based Manipulation of Image Databases*, In Proceedings of Storage and Retrieval for Image and Video Databases II, 2, 185, SPIE, Bellingham, Wash. pp. 34-47, 1994

[12] Salton, G. and Buckley C., *Term-weighting approaches in automatic text retrieval* Information Processing and Management, 1988.

[13] Sclaroff, S., Taycher, L., La Cascia, M., *ImageRover: A Content-based Image Browser for the World Wide Web*, In Proceedings of IEEE Workshop on Content-based Access and Video Libraries, 1997.

[14] Frankel C., Swain M. and Athitsos *Webseer: An Image Search Engine for the World Wide Web*, TR-95-010, Boston University, 1995.

[15] Smith, J. R. and Chang S.-F., *Visualseek: a Fully Automated Content-based Image Query System*, In Proceedings of ACM Multimedia 1996, 1996.

[16] Swain, M. J. and Ballard, D. H., *Color Indexing*, International Journal of Computer Vision, Vol. 7, No. 1, pp. 11-32, 1991.

[17] Gupta, A., *Visual Information Retrieval Technology: A Virage Perspective*, TR 3A, Virage Inc., 1996.

[18] Proceedings of Visual97 Information Systems: The Second International Conference on Visual Information Systems, San Diego, USA, 1997.