# Content-Based Image Retrieval Systems: A Survey

Remco C. Veltkamp, Mirela Tanase Department of Computing Science, Utrecht University email: {Remco.Veltkamp,mirela}@cs.uu.nl

October 28, 2002 This is a revised and extended version of Technical Report UU-CS-2000-34, October 2000

## **1** Introduction

In many areas of commerce, government, academia, and hospitals, large collections of digital images are being created. Many of these collections are the product of digitizing existing collections of analogue photographs, diagrams, drawings, paintings, and prints. Usually, the only way of searching these collections was by keyword indexing, or simply by browsing. Digital images databases however, open the way to content-based searching. In this paper we survey some technical aspects of current content-based image retrieval systems.

A number of other overviews on image database systems, image retrieval, or multimedia information systems have been published, see e.g. [82], [28], [30], [36], [29], [21], [68], [10]. The purpose of *this* survey however, is to provide an overview of the functionality of temporary image retrieval systems in terms of technical aspects: querying, relevance feedback, features, matching measures, indexing data structures, and result presentation. It compares specifc systems, rather than general architectures, and provides a basis for (or defence against) statements like "this or that system already does what your system does". It also is a thorough foundation for claims that most sytems use low level features, and few use high level semantic meaningful features.

A number of keyword-based general WWW search engines allow to indicate that the media type must be images. HotBot (http://hotbot.lycos.com/), and NBCi (http://www.nbci.com/) are examples of these. A number of other general search engines are more specifically for images, such as Yahoo!'s Picture Gallery (http://gallery.yahoo.com/) or the multimedia searcher of Lycos (http://multimedia.lycos.com/), but they are still only keyword based. There are many special image collections on the web that can be searched with a number of alphanumerical keys. For example, ImageFinder (http://sunsite.berkeley.edu/ImageFinder/) provides a list of such collections as a tool to help teachers locate historical photographs from collections around the world.

AltaVista Photofinder (see below) is a search engine that allows content-based image retrieval, both from special collections, and from the Web. In the remainder of this paper, we will give an overview of other such content-based image retrieval systems, both commercial/production systems, and research/demonstration systems.

## 2 Criteria

Many image retrieval systems can be conceptually described by the framework depicted in figure 1. In this article we survey how the user can formulate a query, whether and how relevance feedback is possible, what kind of features are used, how features from query image and data base image are matched, how the retrieval results are presented to the user, and what indexing data structures are used.

The user interface typically consists of a query formulation part and a result presentation part. Specification of which images to retrieve from the database can be done in many ways. One way is to browse



Figure 1: Content-based image retrieval framework.

through the database one by one. Another way is to specify the image in terms of keywords, or in terms of image features that are extracted from the image, such as a color histogram. Yet another way is to provide an image or sketch from which features of the same type must be extracted as for the database images, in order to match these features. A nice taxonomy of interaction models is given in [85]. Relevance feedback is about providing positive or negative feedback about the retrieval result, so that the system can refine the search.

We will consider several classes of features that are used to specify queries: color, texture, shape, spatial layout, and faces. Color features are often easily obtained directly from the pixel intensities, e.g. color histogram over the whole image, over a fixed subimage, or over a segmented region are often used. Although a precise definition of texture is untraceable, the notion of texture generally refers to the presence of a spatial pattern that has some properties of homogeneity. In particular, the homogeneity cannot result from the presence of only a single color in the regions, but requires interaction of various colors.

There is no universal definition of what shape is either. Impressions of shape can be conveyed by color or intensity patterns, or texture, from which a geometrical representation can be derived. This is shown already in Plato's work Meno [67]. (This is one of the so-called Socratic dialogues, where two persons discuss aspects of virtue; to memorialize Socrates, one of the figures is called after him.) In this work, the word 'figure' is used for shape. Socrates' description

"figure is the only existing thing that is found always following color"

does not satisfy Meno, after which Socrates gives a definition in "terms employed in geometric problems":

"figure is limit of solid".

In this paper too we consider shape as something geometrical. Therefore, we consider for example edge orientation over all pixels as texture, but edge orientation at only region contours as shape information. Shape descriptors are diverse, e.g. turning angle functions, deformable templates, algebraic moments, and Fourier coefficients. For an overview of shape matching techniques, see [84].

Indexing is often used as identifying features within an image; with indexing data structures we here mean stuctures to speed up the retrieval of features within image collections.

Spatial layout is about the absolute or relative position of color, texture, or shape information. Higher level features are increasingly more specific, and thus less widely used. However, faces are frequently present in pictures and relatively often used as a feature, so that we tally its use separately.

# 3 Systems

Below we describe a number of content-based image retrieval systems, in alphabetical order. If no *query-ing*, *features*, *matching*, *indexing* data structure, or *result presentation* is mentioned, than it is not applicable to the system (e.g. there is no relevance feedback), or no such information is known to us (e.g. often no information is given about indexing data structures).

## 3.1 ADL

(Alexandria Digital Library)

Developer University of California, Santa Barbara.

**URL** The homepage is at http://www.alexandria.ucsb.edu/adl.html. A walkthrough demo is available at http://alexandria.ucsb.edu/adljigi/tutorials/walkthrough1/. **References** [52], [90].

**Querying** With a map browser the user can interactively pan and zoom a two-dimensional map of the world to locate his areas of interest, and select a query area that must contain or overlap with the database images. Then, the user can select a catalogue of images, set alphanumeric query parameters such as type (e.g. aerial photos, maps, remote sensing images), and then retrieve the images overlapping with the area indicated with the map browser.

**Features** Although it has been reported [51] that images can be searched with texture features, the ADL website does not show this capability. What remains is searching on keywords.

**Matching** When the query is sent to the databases, the query area is compared to the object footprints, according to the functioning matching operator ('contains' or 'overlaps'), and those objects that match are potentially members of the result set for the query.

**Result presentation** The matched images are shown as thumbnails, see figure 2, and their footprints are shown on the map in the map browser.

**Applications** An important focus for ADL's collection is on information supporting basic science, including the Earth and Social Sciences. The image datasets (will) include digital elevation models (DEMs), digital raster graphics (DRGs), scanned aerial photographs, Landsat images, seismic datasets, Sierra Nevada Ecologic Project datasets, and Mojave Ecologic Project datasets.

WORKSPACE (ADL)													
File Options Window Tools			_	_									
All Folders	Title	Туре	Date	Available	Collection								
SADL EXPLORER	🗂 Grapevine Digital	Raster G MAPS	1996-03-27	TIFF									
- 🛄 Query History	🔲 [Scan of] Politica	Imap of tIMAPS	1995-01-01	JPEG	ADL_CATALC								
Query 1	📒 P-44719, [Radar	image of REMOTE-	1994-01-01	TIFF	ADL_CATALC								
	📒 [Image showing	vegetatior REMOTE-	1990-01-01	TIFF	ADL_CATALC								
	🔲 [Scan of] North A	America MAPS	1990-01-01	JPEG	ADL_CATALC								
	4	· · · •••••	1000 01 01	1050	AD1 047417								
CUT COPY PASTE F	OLDER RENAME												
CURRENT FOLDER: Query 1 ITEMS: 10	out of 50												
Thumbnail Imagery		Scan Information—											
		TITLE: <image california="" of="" showing="" vegetation=""/>											
		AVAILABLE AS: TIFF											
		EGIN DATE: 1990-01-01 END DATE: 1992-12-31											
		ID: adl_catalog:1											
and the second					<u> </u>								
	<u>k</u>	•											
METADATA ACCESS RESTORE													
Move and Resize the Frame, or Select A N	Menu Option												

Figure 2: ADL workspace window, showing the query and the resulting thumbnails.

## 3.2 AltaVista Photofinder

**Developer** The AltaVista search engine was originally developed at DEC Research Lab, and is now run by AltaVista Company.

URL http://www.altavista.com/sites/search/simage.

**Features** Similarity is based on visual characteristics such as dominant colors, shapes and textures. No details are given about the exact features.

**Querying** The user first types keywords to search for images tagged with these words. If a retrieved image is shown with a link "similar", the link gives images that are visually similar to the selected image. Similarity is based on visual characteristics such as dominant colors, shapes and textures. The user cannot set the relative weights of these features, but judging from the results, color is the predominant feature.

Indexing The system is built on the VIR Image Engine, see below.

Result presentation The retrieved images are shown as thumbnails, without an explicit order.

#### 3.3 Amore

(Advanced Multimedia Oriented Retrieval Engine)

Developer C & C Research Laboratories NEC USA, Inc.

URL http://www.ccrl.com/amore/.

#### **References** [57], [58].

**Features** The image is segmented into at most eight regions of homogeneous color, and downsized to  $24 \times 24$  pixels. The regions in this picture are directly used for matching [31].

**Querying** The user first selects a category of images. An initial set of images can be selected at random or by keyword. Of these images, visually similar images can be retrieved. The query image can also be specified by its URL. In a research version of the system, sketching a query image was possible. The user can indicate the relative importance of color and shape.

**Matching** First a correspondence between regions in the query and target image is found. Regions corresponding to the same regions in the other image are merged. The shape similarity between two regions is based on the number of pixels of overlap, a kind of template matching. The color similarity between two regions is the distance in HLS space between the uniform region colors [31].

Indexing Indexing is performed by the COIR (Content-Oriented Image Retrieval) system, see [32].

**Result presentation** The retrieved images are shown as thumbnails, without an explicit order, see figure 3. In a research version of the system, result images were displayed as a scatter plot, with shape and color similarity values at the axes, or on a perspective wall [57].



Figure 3: Amore result of similarity retrieval on shape.

**Applications** Amore is used in the art retrieval system Arthur (Art Media and Text Hub and Retrieval System, http://www.isi.edu/cct/arthur/, developed at the Center for Cultural Technology within the Information Sciences Institute of the University of Southern California.

## 3.4 ASSERT

Developer The School of Electrical and Computer Engineering at Purdue University, West Lafayette.

URL The homepage is at http://rvl2.ecn.purdue.edu/~cbirdev/WWW/CBIRmain.html, a demo is at http://rvl2.ecn.purdue.edu/~cbirdev/WEB\_ASSERT/assert.html. References [71]

**Features** A physician delineates a pathology bearing region (PBR) in images. A lobular feature set (LFS) is associated to a lobe or a combination of adjacent lobes. A LFS contains 26 features of PBRs adjacent to the lobular boundary, and 26 features of PBRs interior to it. The features of each PBR include 14 perceptual features that are specific for HRCT lung images, grouped into linear and reticular opacities, nodular apacities, high opacities, and low opacities. In addition, from 255 general purpose features, a sequential forward selection algorithm selected 12: five grey level, five texture, and two shape features. The grey level features are the grey level mean and standard deviation, and bins 6,9, and 16 from a 16-bin histogram. The texture features are contrast, entropy, homogeneity, and cluster tendency, all derived from the grey coocurrence matrix, and the edginess, the ratio of the number edge pixels to the number of pixels in the PBR. The shape features are area, and distance from the PBR region to the enclosing lobular region boundary.

The distribution of the 52-dimensional feature vectors is partitioned into 46 LFS classes.

**Querying** A physician delineates a PBR in the query image, from which the features vector is computed. **Matching** The distance between two feature vectors is the Euclidean distance.

**Indexing** The 52-dimensional feature space of the LFSs is devided into bins. Each LFS is converted into a hash index by the multi-hash method [27]. The index points to the appropriate bin containing pointers to one or more LFS classes.

Result presentation The four best matching images are shown.

**Applications** Assert is a specifically targeted towards retrieval of high resolution computed tomography images of the lung.

#### 3.5 BDLP

(Berkeley Digital Library Project)

Developer University of California, Berkeley.

**URL** The homepage of project is at http://elib.cs.berkeley.edu/, a demo of retrieval from all photos in the collection is at http://elib.cs.berkeley.edu/photos/all.shtml.

#### **References** [13].

**Features** There is a number of alphanumerical keys available for querying: the collection, key words, location, county, and photographer. The colors of each image are quantized into 13 colors bins. Six values are associated with each color bin: the percentage of the image with colors in that bin, and the number of 'very small', 'small', 'medium', 'large', and 'very large' dots of that color found.

**Querying** For content-based search, the user can select 13 colors, and indicate the amount ('any', 'partly', 'mostly') of that color in the picture. Also, for colored regions the user can indicate the size ('any', 'small', 'medium', 'large') and the quantity of regions with that color ('any', 'few', 'some', 'many').

**Matching** Image features are stored as text strings. For example, a picture of a sky with clouds might have a few large white regions, and a large amount of blue, and would have a feature text string "mostly\_blue large\_white\_few". Matching is done by substring matching, e.g. with query string "large\_white%". **Indexing** All features are put into a relational database (the Informix Universal Server database management system).

**Result presentation** The retrieved photos are presented unordered, with id-number, photographer, and collection keys.

**Applications** The collections consist of 23195 images of plants, animals, people, and landscapes, 17000 images from the California Department of Water Resources, Corel Stock photos, and aerial photos of the Sacramento river delta region.

## 3.6 Blobworld

Developer Computer Science Division, University of California, Berkeley.

URL http://elib.cs.berkeley.edu/photos/blobworld/. A demo of Blobworld is available at http://elib.cs.berkeley.edu/photos/blobworld/start.html.

#### References [14].

**Features** The features used for querying are the color, texture, location, and shape of regions (blobs) and of the background. The color is described by a histogram of 218 bins of the color coordinates in Lab-space. Texture is represented by mean contrast and anisotropy over the region, as the 2D coordinate (*contrast*, *contrast* × *anisotropy*). Shape is represented by (approximate) area, eccentricity, and orientation.

**Querying** The user first selects a category, which already limits the search space. In an initial image, the user selects a region (blob), and indicates the importance of the blob ('somewhat', 'very'). Next, the user indicates the importance of the blob's color, texture, location, and shape ('not', 'somewhat', 'very'). More than one regions can be used for querying.

**Matching** To match two color histograms  $h_1$  and  $h_2$ , the quadratic form distance is used:  $d(h_1, h_2) = (h_1 - h_2)^T A(h_1 - h_2)$ , where  $A = (a_{ij})$  is a symmetric matrix of weights representing the similarity between color bins *i* and *j*. The distance between two texture descriptors is the Euclidean distance between their coordinates in representation space. The distance between centroids is the Euclidean distance. The distances are combined into a single final distance.

**Indexing** Rather than actually computing the distances between the full color histogram vectors of length 218 as  $d(h_1, h_2) = (h_1 - h_2)^T A(h_1 - h_2)$ , singular value decomposition (SVD) is used to project the histogram vectors onto a lower-dimensional subspace. The resulting points are indexed by an R\*-tree [6]. **Result presentation** The retrieved images are ranked in linear order, and presented together with the segmented version showing the regions, see figure 4.

Applications The demo on the web provides retrieval from a collection of 10000 Corel stock photos.



Figure 4: Blobworld query result.

## 3.7 CANDID

(Comparison Algorithm for Navigating Digital Image Databases)

Developer Computer Research and Applications Group, Los Alamos National Laboratory, USA.

URL http://public.lanl.gov/kelly/CANDID/index.shtml.

#### References [38].

**Features** Each image is represented by a signature consisting of a weighted sum of Gaussian functions. The color features used are the clusters of spectral bands. The texture features used are the mean, skewness, and kurtosis of each pixel. The feature vectors of all pixels together form a point set in higher-dimensional space. On the basis of the k-means algorithms, clusters are formed. A mean vector and covariance matrix are computed for each cluster, and the associated Gaussian distribution is weighted by the number of elements in the corresponding cluster. The distribution of the feature vectors is now approximated by the weighted sum of the Gaussian distributions.

Querying The user provides a query image.

**Matching** The dissimilarity between two image signatures is based on the normalized Euclidean distance or the inner product of two signatures.

**Result presentation** Each pixel is assigned to one cluster, and each associated Gaussian distribution makes some contribution to the dissimilarity measure. In order to show which parts of the images contribute to the match, each pixel is highlighted depending on the contribution made to the similarity measure.

**Applications** CANDID is used in the retrieval of pulmonary CT images, and multispectral Landsat satellite images.

## 3.8 C-bird

(Content-Based Image Retrieval from Digital libraries)

Developer School of Computing Science, Simon Fraser University, Burnaby, B.C., Canada.

URL http://jupiter.cs.sfu.ca/cbird/.

#### References [46], [47].

**Features** For each collected image, a feature descriptor and a layout descriptor are computed. A feature descriptor is a set of four vectors: a color vector, a most frequent color (MFC) vector, a most frequent orientation (MFO) vector, and a chromaticity vector. A 512-bin RGB histogram is stored in the color vector. The centroids of the regions associated with the 5 most frequent colors form the MFC vector and the centroids of regions of the 5 most frequent edge orientations form the MFO vector. The 36-dimensional chromaticity vector is computed as follows: first, a normalization of each RGB channel is made to obtain illumination invariance, then the 3D color histogram is replaced by a 2D chromaticity histogram. Treating this chromaticity histogram as an image, first a wavelet-based image reduction is applied, then the Discrete Cosine Transform coefficient matrix is built. The chromaticity vector is made of the 36 values of the upper



Figure 5: C-bird query interface and query result.

left corner of the DCT matrix. For search by object model, some geometric data such as the area, the centroid and the eccentricity are computed from color regions associated with each of the MFCs.

The layout descriptor contains a color layout vector and an edge layout vector. To construct these vectors the image is divided into 64 cells, and for each cell the most frequent colors and the number of edges for each orientation are determined.

Also, for images at half and quarter resolution, a feature descriptor like the one described above is stored.

**Querying** The user is presented a grid of consecutive images from the database starting at a random position. To start a query by color histogram or color similarity with illumination invariance, one of the buttons under the selected query image is pressed (see figure 5, left). For a query by color or texture layout, grids are presented for drawing color, texture density and edge orientation layout (see figure 5, right). For a query by color percentage, 5 colors and their percentages are indicated by the user. For a query by object model, the user browses through a selection of query images and makes a choice.

**Matching** The distance between two chromaticity vectors in an illumination invariant color query is the  $L_2$  distance. Texture orientation histograms, as well as color histograms for the full image, are matched by histogram intersection.

The first step in a query by object model is a color localization: color regions for each MFC are extracted and for each region, some geometric data such as the area, the centroid and the eccentricity are computed. After selecting the images in the database that share a number of color regions with the query image, a number of vectors are produced by connecting the centroid of the first MFC region with the centroids of the other MFCs. Analyzing the length of these vectors and the angles between them, a hypothesis regarding the existence of an object at a certain scale and orientation (the difference of angles between centroids of the regions corresponding to the MFCs in the query and database image) is made. This hypothesis is tested in a second step by comparing the texture histogram for each pair of matching regions in the two images. The 2D texture histogram measures orientation (the gradient direction of the edge pixels) and edge separation from the grey level image. Finally, if there is sufficient similarity in their texture between the query object and the area in the database image where the supposed similar object was identified, a shape verification based on the Generalized Hough Transform is performed [5].

**Result presentation** The user can choose the number of rows and columns of the displayed images grid. By clicking on a thumbnail image the user can see some color and texture characteristics of the image (color percentage and layout, texture layout).

#### **3.9 CBVQ**

#### (Content-Based Visual Query)

Developer Image and Advanced Television Lab, Columbia university, NY.

#### URL http://maya.ctr.columbia.edu:8088/cbvq/.

#### References [75].

**Features** The first in a series of systems (which include VisualSEEk, SaFE and WebSEEk) developed by the same team at the Columbia University, CBVQ is the only one that enables queries by texture. First, for each pixel in the image a 9-dimensional texture vector is computed. The components of this vector are the magnitudes of the output of a Haar wavelet filter bank with three iterations on the low frequency. Next, by thresholding the output of each filter, 9 binary images are produced, one for each spatial-frequency subband in the wavelet decomposition. The image pixels are merged into regions of homogeneous texture by using non-linear filtering on each binary image to reduce the noise in the texture content, followed by a sequential labeling algorithm and then a reconstruction of the image by overlapping regions from the binary subband images. Each region is represented by a 9-dimensional binary vector, where each 1 represents a high level of spatial-frequency energy within a particular subband. For each texture region detected, spatial information is also extracted, including region centroid, area, and the dimensions of the minimum bounding rectangle. For each database image, a global color histogram is also computed, and a color segmentation of the image is performed (see VisualSEEk).

**Querying** The system allows queries by example (the user can select one of the displayed random images or gives the URL address of any image on the Web) and direct queries (by constructing a color histogram).



Figure 6: CBVQ interface.

After selecting the query image in a query by example, the user chooses one of the available search methods (see figure 6): color histogram, color composition, texture and histogram manipulation (the query image histogram can be used to construct a new query histogram).

**Matching** Matching by texture regions is most probably done in the same way as color region matching is performed in VisualSEEk.

Indexing See VisualSEEk.

**Result presentation** The best 20 matches are presented to the user in decreasing similarity order, together with the matching score.

Relevance feedback Any of the retrieved images can be chosen as the new query image.

## 3.10 Chabot

Developer Department of Computer Science, University of California, Berkeley, CA, USA.

**URL** http://http.cs.berkeley.edu/~ginger/chabot.html. Chabot has evolved into Cypress (which, surprisingly, seems not to have inherited content based query capability). For a demo of Cypress, see http://elib.cs.berkeley.edu/photos/.

#### References [61].

**Features** One of the early systems, Chabot aimed at combining text based descriptions with image analysis in retrieving images from a collection of photographs of the California Department of Water Resources. The system made use of an existing text description database of the collection, adding other types of textual information for querying such as the shooting date, the picture location, the perspective of the photo. For each image a color histogram containing only 20 bins is computed.

**Querying** The user is presented with a list of search criteria (such as keywords, photographer, film format, shooting date, perspective, location, and colors). The color criterion offers limited options for the user to choose from, such as 'mostly red' or 'some yellow'. The user has the possibility to define concepts, which are combinations of search criteria that the concept satisfies. For example, the concept of 'sunset' is defined as a a combination of keyword ('sunset') and color ('mostly red' or 'mostly orange') criteria.

**Matching** To match a 'mostly...' color criterion, more than 50% of the pixels in an image must have the requested color. For the 'some ...' color criterion, one or two of the 20 colors in the histogram must be qualified as the requested color.

**Indexing** The images and associated data are stored in the database POSTGRES, developed at the University of California, Berkely.

Result presentation Images are shown without specific order.

Applications The database contains 11643 images of California natural resources.

#### 3.11 CHROMA

(Colour Hierarchical Representation Oriented Management Architecture)

Developer School of Computing, Engineering and Technology, University of Sunderland, UK.

URL http://osiris.sund.ac.uk/~cs0sla/.

#### References [43]

**Features** From each image a color dominance vector is generated, of at most length ten. The vector lists the color classes of decreasing dominance. There are ten classes: {uncertain colors like very dark and very bright}, {white}, {grey}, {black}, {red, pink}, {brown, dark yellow, olive}, {yellow, orange, light yellow}, {green, line}, {blue, cyan, aqua, turqoise}, {purple, violet, magenta}. Each pixel of an image tallies for one class, the first vector component is the class with the highest count, etc. If a class does not occur in the image, the vector does not contain this class, and has less than ten components.

A second feature is a  $15 \times 15$  matrix of color classes. First the image is represented at  $15 \times 15$  resolution, which color quantization algorithm is used is not mentioned. The matrix contains the color class of each pixel. This provides a rough description of layout in the image.

**Querying** The user can navigate through a hierarchy, built on the color dominance vectors of all images in the database. The root contains all images. Each branch i below the root contains all images whose color dominance vector starts with i. Each branch j below each node i contains all images whose vector starts with ij, etc. The user can temporarily omit selected color groups.

A second way of querying is with a sketch or query image, whose  $15 \times 15$  matrix is matched with the database images' matrices.

**Matching** The matching score of two images is the Jaccard's coefficient of their matrices, which is the number of matrix elements that are equal, divided by the maximum number, 225.

**Indexing** Except for the navigation, no indexing datastucture is used. The querying with an image or sketch involves matching all database images.

Result presentation The resulting images are presented in linear order of matching score.

#### 3.12 Circus

(Content-based Image Retrieval and Consultation User-centered System)

#### Developer

#### URL http://lcavwww.epfl.ch/~zpecenov/CIRCUS/Demo.html.

#### **References** [66], [65]

**Features** The color feature is a global histogram in Lab space. The texture features are the mean, standard deviation, angular second moment, inverse difference moment, sum average, contrast, correlation, and sum variance, all derived from the cooccurrence matrix of gray values in the direction of  $\pi/4$ . These feature vectors are projected onto a lower dimensional space, derived from the 'feature by image' occurrence matrix  $A = (a_{ij})$ , with  $a_{ij} = l_{ij}g_i$ , where  $l_{ij}$  is the local weighting of feature *i* in image *j*, and  $g_i$  is the global weighting of feature *i*. This is analoguous to the 'term by document' matrix used in Latent Semantic Indexing for text retrieval. Using singular value decomposition (SVD), an orthogonal base for features and images is computed, in which A is expressed as a linear comination. Matrix A is then approximated by the first few terms of this sum.

To speed up the computations, a fast approximation to the decomposition is computed using wavelet packets.

**Querying** From a query image, features are extracted and projected onto the lower-dimensional feature space. Alternatively, color queries can be specified by giving the percentages of each color in the desired image.

**Matching** The distance between two projected feature vectors is the cosine of the angle between the two vectors.



Figure 7: Circus interface.

**Relevance feedback** Users can specify for each query a set of positive and negative examples. The new query consist of the intersection of features from the positive examples, minus the features from the negative examples.

Result presentation The images are shown in linear decreasing order of scoring value.

## 3.13 Compass

(Computer Aided Search System)

Developer Centre for Scientific and Technological Research, Trento, Italy.

URL http://compass.itc.it/.

References [11]

**Features** The color features used are the hue and luminance histograms, and the 2D hue and luminance coocurrence matrices. The texture feature is the histogram of the magnitude of the luminance gradient.

**Querying** The system works with query by example. However, rather than the usual single query image, the query consists of a set of images, see figure 8, which is then sent to possibly more than one image database server.

An alternative is to browse through the database. For this purpose the database is clustered. Key images of all the clusters are projected using multidimensional scaling onto a line.

**Matching** The distance between two feature histograms is the  $L_1$  distance. The distance between two images is a weighted sum of the individual feature distances.

The cloud of query feature vectors in feature space is clustered into a number of query sets  $Q_i$ . The distance  $d(I, Q_i)$  between a database image I and a query set of images  $Q_i$ , is the minimum over all images from the query set.

**Relevance feedback** The user can indicate which result images are relevant, and which are not. The relevant images are added to the query set. When a set of irrelavant images Q' has been specified, the distance d(I, Q, Q') between a database image I and a query set of images Q, given a set of irrelavant images Q', is given by  $d(I,Q)/(d(I,Q)/d(I,Q'))^{\gamma}$ .

**Result presentation** The answers from the multiple servers are then merged and proposed to the user as a single result, shown in decreasing order of similarity score.

#### 3.14 Diogenes

**Developer** Department of EECS, University of Illinois at Chicago. **URL** http://ranger.uta.edu/~alp/diogenes/.

LOTTER REPRESENTATIVES		GORY BAS	MARCE REPORT		
🚓 💑 🐗	-	1	1	**	M
à 🛓 🗿	23		53	W 😸	M
	The charles in 1972 methods with an inte		M	*	0
	marshind max   max   max   max   MMM		96	W 💌	â
	Sectorial Exer- solution to the solution of the		M	M 📾	1
in 1	500 0110 000 00 000 000 000 000 000 000		5		-
se 🔊	200 40 100 1 00002508			1	
S (1)	• me (5) • me(5) • Menter				

Figure 8: Compass interface.

#### References [92]

**Features** Diogenes searches the web for images of persons. The features used are faces in images and text in HTML pages. Some neural network approach is used to detect faces in an image. Once detected, a face is isolated and submitted to a face recognition module based on eigen-faces, which uses a set of known facial images with associated person name for training.

Querying A query is made by submitting a name of a well kown person to the system.

**Matching** The query name is string matched with names in the system index built off-line by a web crawler. An image taken from the web is compared to the training images, yielding a number of distance values. The text of the web page is analyzed to locate person names and to determine their degree of association with the face image. The distance values and degrees of association are input to a classifier, which combines them with a Dempster-Shafer theory of evidence, a generalization of Bayesian theory of subjective probability.

**Result presentation** The images in the database associated with the query name are shown without any explicit order.

**Applications** The system is meant for finding widely known persons on the web, such as celebrities, otherwise there are no images of that person available in the training set.

#### 3.15 DrawSearch

**Developer** Department of Electrical and Electronic Engineering, Technical University of Bari, Italy. **URL** A demo is available at http://deecom03.poliba.it/DrawSearch/DrawSearch.html. **References** [69].

**Features** The features used for querying are color, texture and shape. The image is divided into  $4 \times 4$  subimages and within each subimage, the average color is computed. This results in a 48-length color vector. Shape is represented by means of Fourier descriptors. It is unclear how the contour-based segmentation of the image is done. The components of the shape descriptor are the real part of the lower 100 Fourier coefficients. Texture-based image segmentation makes use of Gaussian Markovian Random Fields.

**Querying** The system has two user interfaces: one that allows queries by color and shape, and the other one allowing queries by texture. In the first subsystem, the user draws a sketch on a canvas, selecting color attributes for the lines and closed regions (see figure 9). In the texture retrieval subsystem, a query is posed



Figure 9: DrawSearch interface.

by selecting a texture area within a database image.

**Matching** In the color/shape subsystem, the similarity between two feature vectors is given by the cosine metric. The similarity score between a query and a database image is calculated as a weighted sum of the distances between the two color vectors and shape descriptors.

Result presentation Images are displayed in decreasing similarity order.

**Relevance feedback** In the retrieval by texture subsystem, the user can improve retrieval results by selecting relevant images from the displayed query results. The other subsystem allows both relevant and nonrelevant images to be marked by the user when reiterating the query. The new query vector is computed by combining the feature vector of the original query Q with the feature vectors of relevant and nonrelevant images:  $Q^{k+1} = Q^k + \delta \sum_{i=1}^{n_{rel}} X_i - \epsilon \sum_{j=1}^{n_{notrel}} Y_j$ , where  $\delta$ ,  $\epsilon$  are positive real weights,  $X_i$  is the feature vector of the *i*th relevant image, and  $Y_j$  of the *j*th nonrelevant image.

#### 3.16 Excalibur Visual RetrievalWare

The Visual RetrievalWare is a software developers kit for building applications for manipulating digital image files and their visual content. The toolkit contains C++ and Java API's for image processing, feature extraction, indexing and content-based retrieval. It also includes sample programs which might be used directly or can serve as templates for building more complex applications. One of these sample programs is the CST (Color, Shape, and Texture) demo.

Developer Excalibur Technologies.

URL http://vrw.excalib.com/. A demo is at http://vrw.excalib.com:8015/cst.

**Features** The CST demo allows queries by example based on HSV color histograms, relative orientation, curvature and contrast of lines in the image, and texture attributes, that measure the flow and roughness in the image.

**Querying** The user first defines the desired visual similarity by specifying the relative importance of the above image attributes, and then selects one of the displayed images as query, see figure 10.

**Result presentation** The images are shown without an explicit ordering.

**Applications** The software has been used in the Image Surfer system, which was used by the Yahoo! and Infoseek WWW search engines.



Figure 10: Excalibur: result of querying with upper left image based on shape alone.

## 3.17 FIDS

(Flexible Image Database System)

**Developer** Department of Computer Science and Engineering, University of Washington, Seattle, WA, USA.

URL http://www.cs.washington.edu/research/imagedatabase/.

#### References [8].

**Features** FIDS uses the following features. The color histogram, and the fraction of flesh colored pixels. The histogram of values after applying the Sobel edge filter. The local binary pattern histogram. The coefficients of the Haar wavelet decomposition. These features are also taken of subimages in a grid, in rows, and columns.

**Querying** The user chooses an image as query image. He can select the feature distance measures, and combine them into an overall distance measure. A result image can be used as next query image.

**Matching** The distances between the histograms are the  $L_1$ -distance. The distance between wavelet coefficients, is some weighted difference. An overall distance can be composed by taking the weighted sum, maximum, or minimum of the individual feature distances, which preserves metric properties.

**Indexing** In a preprocessing step, the distance of all database objects to m reference objects are computed, and stored in a so-called trie structure [25]. At query time, the distances  $d_i$ , i = 1, ..., m of the query to these reference objects are also computed. These both distances and the triangle inequality are used to establish a lower bound on the distance between the query and the database images. The trie structure is used to filter the database images on this lower bound.

**Result presentation** The images can be ordered on their lower bound, so that no exact distances between query and images need to be calculated. Alternatively, the user can select of how many of those the true distance must be calculated.

## 3.18 FIR

#### (Formula Image Retrieval)

**Developer** Developed by Fraunhofer Institute for Computer Graphics, Darmstadt, Germany, in association with Txt Ingegneria Informatica S.P.A. (Italy), Giunti Multimedia Srl (Italy), Epsilon Software (Greece), and Kino TV & Movie Productions S.A. (Greece), as part of the Esprit IV project FORMULA.

#### URL http://www.igd.fhg.de/igd-a7/projects/formula/formula\_e.html. References [88].

**Features** Basic shape and color information of an image are represented using a multiresolution wavelet decomposition. A preprocessing step include a rescaling of the image to a square thumbnail of  $128 \times 128$  pixels and a transformation of the thumbnail colors from the RGB space to Luv space. Next, a non-standard two-dimensional Haar wavelet transform is performed on each color channel individually, followed by a truncation of the resulting coefficients. Only the coefficients larger than a threshold are retained and after rounding to integer values they are stored in a feature vector.

**Querying** A query is submitted by giving an example image.

**Matching** The distance measure between two feature vectors is a weighted Euclidean distance, with different weights for each of the resolution levels in the wavelet decomposition.

## **3.19 FOCUS**

(Fast Object Color-based Query System)

Developer Department of Computer Science, University of Massachusetts, Amherst, MA.

URL http://wagga.cs.umass.edu/~mdas/color\_proj.html. A demo of the system is available at http://cowarie.cs.umass.edu/~colordemo/mdas/demol/phase0.html. References [20].

**Querying** The user can select as query one of the displayed template images, or create a new template by marking a subimage which contains the region of interest.

**Features** Each image is divided in cells of  $100 \times 100$  pixels and for each cell a color histogram in the HSV space, coarsely quantized along the saturation and value axes ( $64 \times 10 \times 10$ ), is computed. The peaks of all local histograms are determined and combined in a list of unique peaks for the whole image by merging multiple copies of the same peak. Also, a frequency table is constructed which, for each color in the HSV space, gives the number of images that have a peak of that color.

The spatial relationships between color regions are represented by means of a spatial proximity graph (SPG) constructed in two phases. First an intermediate SPG is generated, with one node corresponding to each color peak computed for the image cells. Two nodes in this graph are connected if their corresponding peaks are located in the same cell or are located in neighboring cells and have the same color. This graph



Figure 11: Focus. Result of querying with the drawn region of interest.

is then simplified, by unifying all connected nodes of the same color in a single node, and stored using an adjacency matrix representation.

For the query image, a global color histogram is computed and color region relationships are determined at pixel level.

**Matching** The peaks of a query image are subjected to approximate range queries in the increasing order of their corresponding entries in the frequency table. From the resulting lists, the set of images which have peaks matching all query peaks are determined. For the images in this set, a matching score is computed as the the sum of the  $L_1$  distances between each query peak and the matched candidate peak.

To match the SPG of the query image with that of a candidate image, first the candidate SPG is reduced by removing any node whose corresponding peak does not match a query peak. Then it is checked if the query graph appears as a subgraph in the candidate SPG.

**Indexing** The color peaks of the database images are stored in a  $B^+$  tree [40] sorted with hue as the primary key, followed by saturation and value.

**Result presentation** When the user submits a query by clicking on an image, the images are retrieved using the first phase of matching (the images displayed are the images that have all the colors of the query image). By clicking on the 'Refine Results' button, the retrieved images are subjected to the second phase of matching, where the spatial relationships of the matched color regions is analyzed in order to detect a query object in a candidate image.

Applications The database consists of 400 advertisements and 800 color natural images.

#### 3.20 FRIP

(Finding Regions in the Pictures)

Developer Yonsei University, Korea.

URL http://vip.yonsei.ac.kr/frip/.

#### References [39].

**Features** Images are segmented into regions using some circular filter. From each region, color, texture, and shape features are derived. The color feature consists of the average color in RGB space. The texture feature uses the biorthogonal wavelet frame; from each first-level high-pass filtered image, for each region the ratio of x- and y-directional amplitudes is computed. The shape features are eccentricity, and a vector of 12 radius values (centroid-boundary distances), sampled uniformly around the boundary after fixing orientation. In addition, the normalized area, centroid, and length of the major and minor axis are stored.

Querying After segmentation of the query image, the user selects the query region.

**Matching** The distance between two color feature vectors is the city block  $(L_1)$  distance. The distances between two texture ratios, between two normalized areas, and between two eccentricities, are the absolute difference of these two values. The distance between two centroids is the Euclidean  $(L_2)$  distance. The distance between two radius vectors q[1,12] and t[1,12] is the variance of the ratios:  $(1/12)(\sum_{i=1}^{12} |q_i/t_i|/12 - |q_i/t_i|)$ . The final score is a weighted sum of the individual distances.

**Result presentation** The user can specify how many best matches must be retrieved. These images are shown in linear order.

**Relevance feedback** The system retrieves the K nearest neighbors, first using equal weights for all features. The user then marks which retrieved images are relevant, and which are not. The weights are then recomputed on the basis of relative relevance using a probabilistic model.

#### 3.21 ImageFinder

Developer Attrasoft Inc.

**URL** http://attrasoft.com/abm3\_4.html.

**Features** It remains unclear on what features the matching is done.

Querying The querying is done by giving a query image.



Figure 12: ImageFinder training.

**Matching** Matching is done using a Boltzman Machine, a special kind of probabilistic artificial neural network. The network must first be trained on a set of images. This is done by just entering the training images (key images), there is no control over the features on which to train the network, see figure 12. **Result presentation** The result is an HTML page with a list of links to the images found.

#### 3.22 ImageMiner

Developer Technologie-Zentrum Informatik, Univerity of Bremen, Germany.

URL http://www.tzi.de/bv/ImageMinerhtml/.

#### References [41].

**Features** ImageMiner generates content descriptions of images based on color, texture and contours. A color segmentation of the image is done by first computing a color histogram for all grid elements of homogenous size that the image is divided into and then grouping the grid elements according to their most frequent color in the histogram. Bounding rectangles are determined for the grouped grids. For each rectangle, attributes like size, position (both relative to the grid size), associated color and color density (the ratio of the size of color rectangle to the amount of grid elements with that color) are stored.

The texture segmentation is similarly given by the bounding rectangles of the grids grouped according to their similar texture. For representing texture, a statistical approach is adopted. For every grid element, the cooccurrence matrix P is calculated for four orientations (0°, 45°, 90° and 135°). This is an  $N_g \times N_g$ matrix (with  $N_g$  representing the number of distinct gray levels in the quantized image) where  $P_{ij}$  is defined as the joint probability that a pair of neighboring pixels, one with gray-level i and the other with gray-level j, occur in the image under the four specific angular relationships. From each matrix, five statistical features are determined (angular second moment, contrast, correlation, variance and entropy). The average values of these features over the four orientations are the inputs of a neural network trained with a backpropagation algorithm. The network output neurons symbolize seven visual texture properties: bloblikeness, multiareas, planarity, coarseness, regularity, directionality and softness. For every texture rectangle determined, its size, position, the classified texture and the texture density are stored.

The contour-based shape description is obtained by first detecting the edge points using a gradient method and then connecting these points to form contours. For the closed regions detected some shape parameters are extracted (the coordinate of the middle point, the size and the bound coordinates).

The neighborhood relations of the contours and the color and texture regions are represented in a graph. Graph parsing algorithms classify scenes and generate a description with keywords and values. **Querying** SQL-like queries with keywords and values can be posed.

Matching and Indexing Retrieval is performed by the IBM Search Manager text searcher.

**Applications** Videos are made available for retrieval with ImageMiner in two steps. First, shots are extracted from the video by using a histogram based method. More precisely, a new shot is detected when the difference in the intensity histogram of two consecutive frames exceeds a given threshold. Then, a mosaicing-technique is used to generate a single still image for each shot detected in the video sequence. This mosaiced image comprises all the information in a shot and is analyzed by the ImageMiner system as above.

#### 3.23 ImageRETRO

(Image RETrieval by Reduction and Overview)

Developer Department of Computer Science, University of Amsterdam, The Netherlands.

**URL** http://carol.wins.uva.nl/~vendrig/imageretro/. For a demo of the system, look at http://python.wins.uva.nl:6022/.

#### References [86].

**Querying** The user is shown a number of images (called overview), each of them being representative for a part of the image collection. By selecting one of these images, the user selects a part of the database, which is then clustered, and another set of representative images is shown to the user. By this successive filtering, the user obtains a small set of images which can be browsed manually.

**Features** The following features are used in clustering images: a 15-bin hue histogram, the number of colors (the number of bins with more pixels than a threshold, relative to the total number of bins in the hue histogram), the average color (the center of gravity of the circular hue histogram is the bin which gives the lowest average value when taken as a starting point in computing this value; the average value is given by  $avg = (\sum_{j=1}^{15} h_j d_j)/h$ , where  $h_j$  represents the number of pixels in bin j,  $d_j$  is the distance from this bin to the starting point in the circular hue histogram, and  $h = \sum_{j=1}^{15} h_j$ ; the average color is the average value of the center of gravity divided by 15), the color variation (the circular hue histogram is first linearized by using the center of gravity as the center of the new histogram; the standard deviation of this histogram



Figure 13: ImageRETRO after four reduction steps.

normalized by the maximal possible standard deviation gives the color variation), *the grayness* (share of gray valued pixels in the total number of pixels), *the average gray value* (average value of the gray value histogram, see average color), *the average saturation* (the sum of saturation values for each pixel divided by the total number of pixels), *the background proportion* (share of the background in the total number of pixels, where the supposed background is the bin with the highest frequency in the hue or the gray value histogram), *the number of regions* (a region is a collection of adjacent pixels with the same color hue or gray value), *the number of holes* (relative to the number of regions, where a hole is a region surrounded entirely by another region). For all the images in the database, the color features described above are computed off-line and stored in the database.

**Matching** The ranking of the image clusters, employed in the relevance feedback, is straightforward when based on one of the scalar color features. No histogram distance measure is mentioned.

**Relevance feedback** Let  $I_s$  be the image set after s reductions (filterings) and let F denote the set of 10 color features described. The image set is clustered based on an automatically selected feature subset  $F_s$  of F. The images from  $I_s$  are ranked independently for each feature in  $F_s$ , and each such ranking is divided into 4 clusters (corresponding to a reduction of 25%) and each cluster centroid is chosen as the cluster representative. The union of these representatives for all rankings forms the representative set of  $I_s$ , which will be shown to the user for the next reduction. The choice of feature subset  $F_s$  at stage s in the retrieval process is based on statistical analysis. For each feature, the variance of the feature values of all images is computed and  $F_s$  is made of the features with highest variances, that do not highly correlate with each other.

Applications The system was tested with a database of 10000 images, collected from the Web.

#### 3.24 ImageRover

Developer Department of Computer Science, Boston University, MA.

URL http://www.cs.bu.edu/groups/ivc/ImageRover/. For a demo of the system, look at http://www.cs.bu.edu/groups/ivc/ImageRover/demo.html.

#### **References** [70], [83].

**Features** The features used for querying are color and texture orientation. The system computes distributions of color and orientation over 6 subimages (the whole image and 5 subregions: the central and corner



Figure 14: ImageRover query result.

regions). The result is an image index vector made of  $2 \times 6$  subvectors. This dimension is subject to a reduction via a principal component analysis (PCA) for each of the subvector spaces. Image color histograms are computed in the CIE Luv color space and each histogram quantizes the color space into 64 bins. The texture direction distribution is calculated using steerable pyramids. At each of the four levels of the pyramid, texture direction and strength for each pixel is calculated resulting in an orientation histogram, quantized to 16 bins.

**Querying** The user starts a query session by specifying a set of keywords related to the desired images. From the set of images displayed, the user finds and marks one or more images which are similar to what she/he is looking for. Apart from selecting relevant images, the user can deselect one or more of the query images before reiterating a new query. There is no limit to the number of iterations in providing relevance feedback, nor in the number of example images.

**Matching** Based on relevance feedback from the user, the system selects the appropriate  $L_m$  normalized Minkowski metric each time a query is made. The normalization factor  $\mu_m^i$ , employed by the normalized distance  $\tilde{L}_m(x_i, y_i) = L_m(x_i, y_i)/\mu_m^i$ , with  $x_i$  and  $y_i$  being two subvectors of the image index vectors Xand Y, is the expected value over the entire database:  $\mu_m^i = E[\tilde{L}_m(x_i, y_i)]$ . It is allowed to use metrics of different orders  $m_i$  for each of the image index subvectors. Thus, if S is a set of relevant images indicated by the user, the appropriate value for m of the *i*-th subvector is chosen so as to minimize the distance between relevant images:  $m_i = \arg \min_m E[\tilde{L}_m(p_i, q_i)]$ , over all  $P, Q \in S$ .

Then the k-nearest neighbor search of the image index vector uses the following weighted distance metric:  $d(X,Y) = (w_1, \ldots, w_n) (\tilde{L}_{m_1}(x_1, y_1), \ldots, \tilde{L}_{m_n}(x_n, y_n))^T$  where X, Y are the image index vectors and  $w_i$  are relevance weights.

**Indexing** An optimized kD tree is used and the search algorithm in this structure employs an approximation factor. The user can select through the search accuracy button one of the three values for this approximation factor.

**Result presentation** Images similar to the query images are shown in decreasing similarity order. Their number is chosen by the user. Each displayed thumbnail image is a hypertext link to the original image, allowing the user to retrieve the desired image from its home WWW site.

**Relevance feedback** The user indicates the more relevant images, after which the system selects the appropriate  $\tilde{L}_m$ , see above.

Applications Image Rover is an image search engine for the WWW.

#### 3.25 ImageScape

Developer Department of Computer Science, Leiden University, The Netherlands.

URL http://www.wi.leidenuniv.nl/home/lim/image.scape.html. A demo is available at http://ind134a.wi.leidenuniv.nl:2001/new2/imagesearch.demo.html. References [44], [12].

**Querying** Using the sketch interface, the user can draw an outline of the desired image. For semantic querying, the user brings icons on a canvas that represent the objects/concepts he is looking for, at the desired position in the image. Examples of object/concept categories include human faces, stone or sand, water, sky, tree or grass, points and lines (see figure 15).

**Features** Edge maps of the images collected by Web crawlers are obtained using the Sobel operator and a Gaussian blurring filter. A frequency histogram of the  $3 \times 3$  binary pixel patterns occurring in the edge image, which is called trigram vector, is computed for all images. This vector is subjected to a dimensionality reduction using a band-pass filter (see LCPD below). Various other features, used in object matching, are taken at pixel level: color, Laplacian, gradient magnitude, local binary patterns, invariant moments and Fourier descriptors.

**Matching** The first step of the object matching process uses the  $L_1$  distance on the trigram vectors to retrieve the top 1% matches from the entire database. Among these, 20 matches are selected in a second step, a 20 × 20 template matching, using the most informative pixels to minimize the misdetection rate. These pixels are found as follows. For each object, a large set of positive and negative examples are used in finding the set of 256 pixels with the greatest discriminatory power, by maximazing the Kullback relative



Figure 15: ImageScape query built with icons.

information combined with a Markov random field.

## **3.26 iPURE**

(Perceptual and User-friendly Retrieval of Images)

Developer IBM India Research Lab, New Delhi, India.

**URL** http://www.research.ibm.com/irl/projects/mediamining/index.html.

## **References** [1].

Features Images are segmented using the Mean Shift Algorithm. Of each region, color, texture, and shape features are extracted, as well as spatial lay-out. The color features are the average Luv color. Texture features that are used are the coefficients of the Wold decomposition of the image viewed as a random field. The shape features are size, orientation axes, and Fourier descriptor. The spatial lay-out is captured by the centroid, the minimum bounding box, and contiguity.

Querying The user starts the retrieval session with a query image which is sent to the server, which sents back a segmented image. The user can select one or more segments

Matching Individual segments are matched by computing a weighted Euclidean distance on the feature vectors. Then the contiguity amongst the retrieved segments are matched against those of the query segments.

Relevance feedback The user can give relevance feedback modified versions of the retrieved images. The modifications are change of color (bit flipping in RGB domain), texture (Wold frequencies), shape (changes of size and the highest Fourier coefficients), and spatial lay-out (position and orientation). If the user finds the modified images also relevant, it means that the particular feature has low relevance. The new query is defined as the mean of the feature vectors of the relevant images.

#### 3.27 Jacob

(Just A COntent Based query system for video databases)

Developer Computer Science & Artificial Intelligence Lab, University of Palermo, Italy.

URL http://www.csai.unipa.it:80/research/projects/jacob/. A demo is available at http://www.csai.unipa.it:80/research/projects/jacob/jacob\_demos.html. **References** [15].

Features The system makes queries based on color and texture features. Color is represented by means of a histogram in the RGB space. Texture features used are two measures extracted from the grey-level cooccurrence matrix, the maximum probability  $max_{m,n}f(m, n, r, \theta)$ , and the uniformity  $\sum_{m}\sum_{n}(f(m, n, r, \theta))^2$ . An 8-dimensional vector is obtained by computing the joint probability  $f(m, n, r, \theta)$  for distance r = 1

and orientation  $\theta = -45^{\circ}, 0^{\circ}, 45^{\circ}$ , and  $90^{\circ}$ . Another vector is composed of edge density measures (the fraction of edge pixels in the total number of pixels in the image) along four directions,  $-45^{\circ}, 0^{\circ}, 45^{\circ}$ , and  $90^{\circ}$ .

**Querying** The queries may be direct or by example. A direct query is made by inserting a few values representing the color histogram and/or the texture features. For a query by example, the user must give an image.

**Matching** Two color histograms  $H_1, H_2$  are compared using the following distance measure:  $d(H_1, H_2) = \sum_{y \in S} \sum_{x \in I(y)} |H_1(y) - H_2(x)| s(y, x)$ , where S is the reduced RGB space, I(y) is a small interval centered on the point y and s is a color similarity function. The distance measure used for the texture feature vectors is not mentioned. When performing a query, the user choose a value between 0 and 1 to indicate the relative importance of a feature with respect to the other one. As a result the two distances computed are weighted in a global similarity measure.

**Result presentation** The system returns the best matching frames in a similarity order. The number of returned frames is chosen by the user.

Applications The matching of images is used for querying video databases.

#### 3.28 KIWI

(Key-points Indexing Web Interface)

Developer INSA Lyon, France.

URL http://telesun.insa-lyon.fr/kiwi/.

#### **References** [48], [49].

**Features** First, key points are dectected in the image, using some wavelet-based salient point detector [49]. From the collection of all pixels in the  $3 \times 3$  neighborhoods around the key point, histograms are built of the color moments mean, variance, and skewness, for the three color channels R, G, and B. A number of intensity surface derivative invariants are computed and stored in a 2D histogram, built over all pairs of key points. Gabor filters are applied to regions around the key points, and for each combination of 3 scales and 8 orientations, a 2D histogram of amplitude values is built over pairs of key points. Relative spatial lay-out of key points is encoded by building two 2D histogram on values  $(R, \alpha)$  and  $(n, \alpha)$ , with R = min(|AB|, |AC|)/max(|AB|, |AC|), and  $\alpha = \arccos(AB + AC/|AB| |AC|)$ , where B and C are



Figure 16: KIWI web interface.

successive k-nearest neighbors of A, and n is the rank of B among them. (Note that this relative spatial layout of key points is called a shape feature in this system.)

**Querying** The user can specify a query image by selecting an image from the database, or by providing an URL of an image.

**Matching** The color feature vectors are compared with the Euclidean distance. After normalization, the 2D histograms are compared using the Bhattacharyya distance:  $d(H_1, H_2) = -log \sum_{i,j} \sqrt{H_1(i, j)H_2(i, j)}$ . After a normalization of the distribution of distance values from the individual features, the similarity values are sorted and then weighted averaged.

Result presentation The retrieved images are presented in linear order with respect to the final score.

#### 3.29 LCPD

(Leiden 19th Century Portrait Database)

Developer Department of Computer Science, Leiden University, The Netherlands.

URL http://ind156b.wi.leidenuniv.nl:2000/. A demo of the system is available at site http://ind156b.wi.leidenuniv.nl:2000/cgi-bin/image\_demo6.0.pl.

#### References [45].

**Features** The user has several choices in selecting the feature vector, the pixel value domain used for computing this vector, and the resolution level. There are three options for the pixel domain: the intensity image, the gradient image (obtained by Sobel operators) and the thresholded gradient image. One feature vector is the horizontal/vertical projection vector. For an image with  $m \times n$  pixels, this vector has m + n components computed as the average of the row/column pixel values in the selected space. A second feature is the trigram vector, a frequency histogram of the  $3 \times 3$  binary pixels patterns in the thresholded gradient image. This 512-length vector can be subjected to a dimensionality reduction and to a component weighting scheme (low weights are applied on either end of the sorted frequency range). A way of reducing the length of the trigram vector is by forming groups of rotation, mirroring and/or intensity invariant binary patterns. By using the RM (Rotation, Mirroring) group, the dimension of the feature vector is reduced to 102 and by forming RIM (Rotation, Intensity, Mirroring) groups to 51. Another method used consists of suppressing the contribution of the black and white patterns (which are among the most common patterns) and the rare patterns. This is called a bandpass filter. A Karhunen-Loeve Transform can also be used for



Figure 17: LCPD, query image at upper left.

feature vector length reduction.

A similar vector can be constructed in the intensity space. In this case, the feature vector is computed by first thresholding the 8 neighbors of each pixel with its intensity value and counting the number of occurrences of each of the  $2^7$  possible patterns of these 8 pixels (the center pixel is not part of the pattern). This results in an 256-length local binary pattern vector (LBP), which can also be subjected to dimensionality reduction.

**Querying** Querying is done by example. The user first selects a search method (different combinations of pixel domain, feature vector, dimensionality reduction scheme employed and resolution) from the displayed list.

**Matching** The similarity between two feature vectors (projections, trigram or LBP vectors) is given by the  $L_1$  distance. If two images have different sizes, the similarity score between two projection vectors is given by the minimum  $L_1$  distance over all possible x-/y-translations of one image over the other. Another similarity measure is given by the magnitude of the the average pixel to pixel difference in the intensity or gradient space.

**Indexing** The POSTGRES database stores for each image the best matches for 30 search methods.

**Result presentation** Retrieved images are presented in decreasing similarity order, see figure 17).

**Applications** The system is used to find near copies of Dutch carte de visite studio portraits from a database of 16505 images of photographs taken from 1860 till 1914.

#### Performance

http://ind156b.wi.leidenuniv.nl:2000/cgi-bin/performance/m\_pagela.pl provides information about performance.

#### 3.30 MARS

(Multimedia Analysis and Retrieval System)

**Developer** Department of Computer Science, University of Illinois at Urbana-Champaign, further developed at Department of Information and Computer Science, University of California at Irvine, CA.

URL http://www-db.ics.uci.edu/pages/research/mars.shtml. Demos are available at location http://www-db.ics.uci.edu/pages/demos/index.shtml.

#### References [62].

**Features** The system supports queries on combinations of low-level features (color, texture, shape) and textual descriptions. Color is represented using a 2D histogram over the HS coordinates of the HSV space. Texture is represented by two histograms, one measuring the coarseness and the other one the directionality of the image, and one scalar defining the contrast. In order to extract the color/texture layout, the image is divided into  $5 \times 5$  subimages. For each subimage a color histogram is computed. For the texture of a subimage, a vector based on wavelet coefficients is used. The object in an image is segmented out in two phases. First, a k-means clustering method in the color-texture space is applied, then the regions detected are grouped by an attraction based method. This consists of choosing a number of attractor regions and associating each region with the attractor that has the largest attraction to it. The attraction between two regions, i and j, is defined as  $F_{ij} = M_i M_j / d_{ij}^2$ , where  $M_i, M_j$  are the sizes of the two regions and  $d_{ij}$  is the Euclidean distance between the two regions in the spatial-color-texture space. In the MARS system, five attractors are used: one for each corner of the image (background attractors) and one in the center of the image (the objects attractor). This is consistent with the fact that their database consists of images of single objects. The shape of the boundary of the extracted object is represented by means of Fourier Descriptors (FD).

**Querying** Complex queries can be formulated using boolean operators. The desired features can be specified either by example (pointing an image database that has such a property) or direct (for example, by choosing colors from a palette or textures from an available set of patterns).

**Matching** The similarity distance between two color histograms is computed by histogram intersection. The similarity between two textures of the whole image is determined by a weighted sum of the Euclidean distance between contrasts and the histogram intersection distances of the other two components, after a normalization of the three similarities. For computing the texture similarity between two corresponding



Figure 18: Mars shape query with drawn polygon.

subimages, the Euclidean distance between the vector representations is used. A weighted sum of the  $5 \times 5$  color/texture similarities is used to compute the color/texture layout distance between two images. The similarity measure between two FD shape representations is a weighted sum of the standard deviations of  $ratio(k) = M_2(k)/M_1(k)$  and  $shift(k) = \theta_2(k) - \theta_1(k) - \psi$ ,  $k = -N_c, \ldots, N_c$ , where  $M_i(k)$  and  $\theta_i(k)$  are the magnitude and the phase angle of the FD coefficients,  $\psi$  is the difference of the major axis orientations of the two shapes and  $N_c$  is the number of FD coefficients.

Each query has a query tree associated. In a query tree, the leaves represent the feature vectors (the terms of the boolean expression defining the query) while the internal nodes correspond to boolean operators or more complex terms indicating a query by object. Individual queries on each of the query terms are made. The tree is evaluated bottom-up: each internal node receives from each child a list of ranked images and combines these lists, after a normalization process, according to the weights on the parent-child links. **Indexing** There is no information about indexing data structures used for queries. A new version of the system, WebMARS, is developed where the feature vectors are indexed using hybrid trees, which combine aspects of several indexing trees.

Result presentation Images are listed in order of decreasing similarity.

**Relevance feedback** Initially, the weights of the edges in the query tree are equal for the children of the same parent and their sum is 1. Based on the relevant images chosen by the user from the query result list, a tree reweighting process takes place.

**Applications** The database consists of images of ancient African artifacts from the Fowler Museum of Cultural History at UCLA.

#### 3.31 MetaSEEk

Developer Image and Advanced Television Lab, Columbia University, NY, USA.

URL http://www.ctr.columbia.edu/metaseek/.

#### References [7].

**Features** Content-based retrieval can be done on the basis of color and texture. MetaSEEk has a color or texture matching and indexing of its own to cluster query images in a local performance database, used to select target serach engines. The actual image matching is forwarded to QBIC, VIR Image Engine, WebSEEk, and the VisualSEEk retrieval engines.

**Querying** The user can select a category, provide a key word, provide a URL of an image, or select a shown image. Upon receiving a query, the dispatcher consults the performance database at the MetaSEEk site. This database contains performance scores of past query successes and failures of each supported search option. This is used to select the target search engines to be queried.

**Indexing** Query images in the performance database are clustered into several classes on the basis of color, textures, and the combination of both. When the user issues a query with a new image for which no performance scores are available, the system downloads the image, and matches it to the corresponding clusters in order to obtain a list of the most similar clusters. Selected images from the few closest clusters are presented to the user, who can choose one. In this way, new queries are related to the performance of past ones for which performance information is available.

**Matching** Color and texture are extracted locally by MetaSEEk for the clustering. Color similarity is computed by calculating the color histogram of an image. Texture is computed by measuring the coarseness, contrast, and presence/absence of directionality of an image. The distance between two feature vectors is the Euclidean distance.

Result presentation The display component merges and ranks the results form each search option.



Figure 19: MetaSEEk results and relevance feedback.

United	2	4.4 .	0.2	15 2
2.0	Ca Men Jr I	e Wegteng 6-Orres Stor-Y	tow Coler (In	wrighting share   Edge III III
	340	lick on an imag	e sa dina sinalise s	niget.
AL V			AL	
Transfe		N.M.		
4			1 des	非影
11		10- J	11	1 8

Figure 20: MIDSS.

**Relevance feedback** The user can indicate to like or dislike each of the returned images, see figure 19. This information is used to modify the corresponding entries in the performance database.

## 3.32 MIDSS

(Multi-Resolution Image Database Search System)
Developer Electrical and Computer Engineering, Purdue University, Indiana.
URL http://albrecht.ecn.purdue.edu/~jauyuen/webfind.html.
References [17].
Features
Querying and Result presentation
Matching
Relevance feedback
Indexing
Applications

## 3.33 MIR

(Multimodal Information Retrieval System)

**Developer** Center of Excellence for Document Analysis and Recognition, University at Buffalo, NY, USA. **URL** http://www.cedar.buffalo.edu/MMIR/. A demo of the system is available at webaddress http://www.cedar.Buffalo.EDU/MMIR/demo/index.html. **References** [78], [79].

**Features** The MIR system combines various techniques from text processing and image processing in an attempt to derive semantic descriptions of images. By employing natural language processing (NLP) techniques in analyzing image captions, information about the picture's content is derived. This includes whether there are people in the picture, their names, location and time of the photograph, spatial relationships of the people in the image as well as other visual characteristics (attributes that can assist in face recognition such as gender, hair color, beards, mustaches and glasses). Also, statistical text indexing techniques are used in capturing the general context of the image (such as indoor versus outdoor). The pictures in which the presence of people is detected by NLP, are subjected to a face detection to verify this hypothesis. The face detection algorithm uses a three-contour model representing hair line, and left and right



Figure 21: MIR query result.

outlines of the face. After extracting edges at different image resolutions by means of a wavelet transform, a graph-matching generates possible face candidates in the image. After the face detection, the face areas are cropped out, and for the rest of the image a color histogram is computed. For the scenery pictures (where no face is detected in the NLP analysis) a color correlation histogram is computed. If  $\{i, j, k\}$  is a color triple in the  $2 \times 2 \times 2$  quantized HVS space, then  $h_{ijk}^l$  denotes the number of occurrences of three pixels of these colors as the vertices of a isosceles right triangle with the smaller sides of length l. The color correlation histogram  $f_{ijk}^l$  is  $h_{ijk}^l/(4 * h_i)$ , where  $h_i$  is the  $i^{th}$  bin of the traditional color histogram. Querying A query formulation can include a text string, an image, and a topic (selected from a predefined set containing sports, politics, entertainment etc.). The user can also indicate the relative importance of text versus image content, as well as of background versus foreground in an image containing people (the foreground).

**Matching** The similarity between a query and a database image is a weighted sum of the similarities between different sources of information (text-based and content-based) extracted from the image. Two color correlation histograms are matched using the Euclidean distance.

**Applications** The system retrieves pictures of people and/or similar scenery in various contexts. Three databases were used in testing the system. The first one consisted of approximately 5000 images with accompanying text provided by United Press International. The second one provided by Kodak consists of consumer photos accompanied by speech annotations. The third one consists of multimodal documents downloaded from the Web.

## 3.34 MUSE

(Multimedia Search) Developer Computer Science and Engineering, Florida Atlantic University. URL . References Features Querying Matching Result presentation Relevance feedback Indexing Applications

#### **3.35 NETRA**

**Developer** Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA.

**URL** http://maya.ece.ucsb.edu/Netra/. A demo of the system is available at webaddress http://maya.ece.ucsb.edu/Netra/netra.html.

#### **References** [50], [51].

**Features** Images in the database are segmented into regions of homogeneous color. Of those regions, the following features are extracted: color, texture, shape, and spatial location.

On the basis of a training set of images, the RGB color space is quantized, and represented by a color codebook of 256 colors, the centroids of the quantization cells. The colors of an image region are also quantized, giving a color feature vector  $f_c = (c_0, p_0, \ldots, c_n, p_n)$ , with  $c_i$  the index into the color code book, and  $p_i$  the fraction of that color in the region,  $p_0 + \ldots + p_n = 1$ . The number n is the number of colors used to represent the region, which is different for each region.

Texture is represented by a feature vector  $f_t$  containing the normalized mean and standard deviation of a series of Gabor wavelet transforms of the image:  $f_t = (\mu_{0,0}, \ldots, \mu_{s,k}, \sigma_{0,0}, \ldots, \sigma_{s,k})$ , with s the number of scales, and k the number of directions.

There are three feature vectors used to represent the shape of regions. The first,  $f_K$ , is based on the curvature function of the contour, giving the curvature at each point on the contour. The second,  $f_R$  is based on the centroid distance function, giving at each contour point the distance to the centroid of the region. The third,  $f_Z$ , is the complex coordinate function, representing each contour point as a complex number with real component equal to the x-coordinate, and the imaginary component equal to the y-coordinate. On 64 samples of each of these functions, the fast Fourier transform (FFT) is applied, of which the real (amplitude) component of the coefficients is used, the numbers  $F_{-31}, \ldots, F_{32}$ . The feature vectors are as follows:  $f_K = (|F_1|, \ldots, |F_{32}|), f_R = (|F_1|, \ldots, |F_32|)/|F_0|, f_Z = (|F_{-31}|, \ldots, |F_{-1}|, |F_2|, \ldots, |F_{32}|)/|F_1|$ . Querying There are 2,500 images from the Corel photo collection, organized in 25 categories, with 100 images in each category. You can select any one of them as the query image. All images in the database

have been segmented into homogeneous regions. You can click on one of the regions and select one of the



Figure 22: NETRA. Result of querying on shape with the complex description.

four image attribute color, spatial location, texture, and shape. Instead of using an image example, you can also directly specify the color and spatial location. The spatial location querying tool utilizes two bounding boxes to define the area of interest. The inner box is used to define the preferred area, and the box outside is used to constrain the objects to be within this area. Thus, if the object has any its bodies exceeding this outside box, they will not be considered.

**Matching** Consider two color feature vectors,  $f_c^A$  of region A, and  $f_c^B$  of region B. For each color  $c_i$  in  $f_c^A$ , the closest color  $c_k^B$  in  $f_c^B$  is found, and the distance  $d(c_i^A, f_c^B)$  is calculated as the weighted Euclidean distance in RGB space:  $d(c_i^A, f_c^B) = |p_i^A - p_k^B| d(c_i^A, c_k^B)$ . The distance between the two color feature vectors is now  $\sum_{i=0}^{n^A} d(c_i^A, f_c^B) + \sum_{i=0}^{n^B} d(c_i^B, f_c^A)$ . The distance between two texture feature vectors is the  $L_1$ -distance. The distance between two shape feature vectors is the Euclidean distance.

Indexing is based on the SS-tree [91]. Color, texture, and shape are indexed separately. The first feature the user specifies is used to retrieve about 100 candidates. Then this feature and the possible other features together are used to order the retrieval result.

Result presentation The matched images are linearly ordered, see figure 22.

Applications An initial prototype of NETRA is used in ADL (see above) to search on texture.

#### 3.36 PARISS

(Panoramic, Adaptive and Reconfigurable Interface for Similarity Search) Developer URL . References Features Querying Matching Result presentation Relevance feedback Indexing Applications

#### 3.37 Photobook

Developer Vision and Modeling Group, MIT Media Laboratory, Cambridge, MA.



Figure 23: Photobook result, query at far left.

**URL** http://vismod.www.media.mit.edu/vismod/demos/photobook/index.html.At http://vismod.www.media.mit.edu/cgi-bin/tpminka/query?vistex,,,10 a demo is available.

References [64].

**Features** Photobook implements three different approaches to constructing image representations for querying purposes, each for a specific type of image content: faces, 2D shapes and texture images. The first two representations are similar in the way that they offer a description relative to an average of a few prototypes by using the eigenvectors of a covariance matrix as an orthogonal coordinate system of the image space. First a prepocessing step is done in order to normalize the input image for position, scale and orientation. Given a set of training images,  $\Gamma_1, \Gamma_2, \ldots, \Gamma_M$ , where  $\Gamma_i$  is a  $n \times n$  array of intensity values, their variation from the average,  $\Psi = \frac{1}{M} \sum_{i=1}^{M} \Gamma_i$ , is given by  $\Phi_i = \Gamma_i - \Psi$ . This set of vectors is then subjected to the Karhunen-Loève expansion, the result being a set of M eigenvectors  $u_k$  and eigenvalues  $\lambda_k$  of the covariance matrix  $C = \frac{1}{M} \sum_{i=1}^{M} \Phi_i \Phi_i^T$ . In representing a new image region,  $\Gamma$ , only  $M_1 < M$  eigenvectors with the largest eigenvalues are used, thus the point in the eigenimage space corresponding to the new image is  $\Omega = (\omega_1, \omega_2, \ldots, \omega_{M_1})$ , where  $\omega_k = u_k^T (\Gamma - \Psi)$ ,  $k = 1, \ldots, M_1 < M$ .

In a texture description, an image is viewed as a homogeneous 2D discrete random field, which by means of a Wold decomposition, is expressed as the sum of three orthogonal components. These components correspond to periodicity, directionality and randomness.

In creating a shape description, first a silhouette is extracted and a number of feature points on this are chosen (such as corners and high-curvature points). This feature points are then used as nodes in building a finite element model of the shape. Solving the following eigenvalue problem  $K\phi_i = \omega_i^2 M \phi_i$ , where M and K are the mass and stiffness matrices, respectively, the modes of the model are computed. These are the eigenvectors,  $\phi_i$ , which are next used for determining a feature point correspondence between this new shape and some average shape.

**Querying** To perform a query, the user selects some images from the grid of still images displayed and/or enters an annotation filter. From the images displayed, the user can select another query images and reiterate the search.

**Matching** The distance between two eigenimage representations,  $\Omega_1$  and  $\Omega_2$ , is  $\epsilon_{ij}^2 = ||\Omega_i - \Omega_j||^2$ . Two shapes are compared by calculating the amount of strain energy needed to deform one shape to match the other.

**Indexing** Prior to any database search, a few prototypes that span the image category are selected. For any image in the database, its distance to the average of the prototypes is computed and stored for future database search. At query time, the distance of the query image to the average is computed and the database is reordered according to this.

**Result presentation** Images in the database are sorted by similarity with the query images and presented to the user page by page.

**Applications** The face recognition technology of Photobook has been used by Viisage Technology in a FaceID package, which is used in several US police departments.

#### 3.38 Picasso

Developer Visual Information Processing Lab, University of Florence, Italy.

**URL** http://viplab.dsi.unifi.it/PICASSO. A web demo of the system can be evaluated at http://vesna.dsi.unifi.it/~picasso/picasso/.

#### **References** [9], [63].

**Features** The Picasso system supports queries based on shape, color regions and their spatial relationships. The system exploits a pyramidal color segmentation of the database images, each level of the pyramid corresponding to a resolution level of the segmentation. At the lowest level, each pixel in the image is identified with one region, while at the highest level of the pyramid the entire image is represented by a single region. Image regions at level n are determined by iteratively merging adjacent regions from level n - 1, minimizing the sum of all region energies. The energy  $E_R$  of a region R is given by  $E_R = \alpha 1/A(R) + \beta D(R) + \gamma \sum_{R_i \in N(R)} 1/D(R \cup R_i)$ , where A(R) is the region's area, D(R) is a measure of

the uniformity in color of the pixels belonging to the region R, N(R) denotes the set of regions adjacent to R and  $\alpha$ ,  $\beta$  and  $\gamma$  are positive real weights. This way, at the end of the segmentation process, each image is associated with N segmented images  $I_n$ ,  $n = 1 \dots N$ , which are represented using a multilayered graph G. The nodes of G are the color regions of the N segmented images. There are two types of connections between these nodes: *intralevel* links between the nodes corresponding to adjacent regions in a segmented image  $I_n$  and *interlevel* links between a node correspondig to a region  $R_n^k$  of  $I_n$  and the nodes corresponding to regions of  $I_{n-1}$  that merged to form  $R_n^k$ . The description of a color region (graph node) include: a binary 187-dimensional color vector, the position of the region centroid, the region area (the fraction of region pixels in the total number of image pixels) and the region shape (the region is approximated by its best fit ellipse and the features used are the elongation and the major axis orientation). One property of the color vectors that helps in indexing and matching is that the color vector of a region Ris computed as the union of the color vectors of the regions that merged to form R.

In order to allow queries by shape, the objects of interest in each image are bounded with their minimum enclosing rectangle (MER) during the database population step. Edge images ( $128 \times 128$  in size) of these rectangular areas are extracted through a Canny edge detection. Also, spatial relationships between objects' MER are represented using 2D strings [16]. The query contours drawn by the user are modeled by B-spline functions.

**Querying** The system interface allows the user to select the type of search: by color regions or by shape. In the first case, regions can be indicated in two ways: either by drawing contours, which are then filled with colors chosen from a palette, or by loading a database image and tracing the contour of some relevant regions in this image. The user can also adjust the relevance of the region attributes (position, area, elongation and orientation). A query by shape is based on user drawn sketches, while for a query by texture patterns, the user has to pick an image from the database.

**Matching** In a query by color regions, after a fast selection of the database images that contain all the colors of the query (see *Indexing*), the pyramidal structure of each candidate image is analyzed from top to bottom to find the best matching region for each query region. The matching score between a query region and a candidate image region is given by a weighted sum of distances between the computed region attributes (color, region centroid's position, area and shape). The similarity score between the query image and a candidate image is obtained by summing all the scores of the matched query regions.

In a shape based query, first images are filtered according to the spatial relationships and positions of the delimited MERs, based on the 2D string representation. 1D elastic matching is applied to the images that have passed this filtering step. If the sketch contains k contours, the systems warps each contour over the candidate image's shape located in the same relative position as the query contour. The similarity score between the deformed contour and the image object takes into account both the match M between the deformed contour and the amount of elastic energy E used to warp the query contour. A gradient descent technique is used in minimizing E - M.

**Indexing** In the color region queries, an expensive matching of the query against all database images is avoided by exploiting the color vectors. Because the color vector of the node in the highest level of the pyramidal color region representation includes colors of all the regions in the pyramid, a color index file is constructed. This contains the color vectors of the highest node of the pyramidal structure of each image and is used in the filtering step of the matching process, when only the images that contain all the colors of the query are selected for further inspection.

**Result presentation** The query results are presented to the user in decreasing similarity order.

**Relevance feedback** A query by sketch can be refined, for example by choosing one of the retrieved images and iterating a query by grey level intensity surface.

**Applications** The system is used for the electronic cataloging of paintings and sculptures of museums and galleries in the area of central Italy.

#### 3.39 PicHunter

**Developer** NEC Research Institute, Princeton, NJ, USA. **References** [19].

**Features** The current version of the systems uses color histogram and color spatial distribution along with hidden textual annotations. Besides a 64-bin HSV histogram, two other vectors – a 256-length HSV color autocorrellogram (CORR) and a 128-length RGB color-coherence vector (CCV) – are describing the color content of an image. The first 64 components of CORR represent the number of pixels of a particular color from the 64-quantized HSV color space having neighbors of the same color at distance 1. The rest of the vector is defined in the same way for distances 3, 5 and 7. CCV is made of 64 coherence pairs, each pair giving the number of coherent, and incoherent pixels of a particular discretized color in the RGB space. In order to classify pixels in one of the two categories, the image is first blurred slightly by replacing each pixel value with the average value of the pixels in its  $3 \times 3$  neighborhood. Then, pixels are grouped in connected components. A pixel will be classified as coherent if the connected component it belongs to is larger than a fixed threshold. Keywords, selected from a set of 138 words, are associated with each image in the database population step and are represented as a boolean vector.

Querying Retrieval is done with query by example.

**Matching** The distance between individual features (color vectors or annotation lists represented as binary vectors) is the  $L_1$  distance. These distances are scaled and combined in a global distance. The scaling factors are computed by maximizing the probability of a training set.

**Relevance feedback** PicHunter implements a probabilistic relevance feedback mechanism, which tries to predict the target image the user wants based on his actions (the images he selects as similar to the target in each iteration of a query session). A vector is used for retaining each image's probability of being the target. This vector is updated at each iteration of the relevance feedback, based on the history of the session (images displayed by the system and user's actions in previous iterations). The updating formula is based on Bayes' rule. If the *n* database images are noted  $T_j$ , j = 1, ..., n, and the history of the session through iteration *t* is denoted  $H_t = \{D_1, A_1, D_2, A_2, ..., D_t, A_t\}$ , with  $D_j$  and  $A_j$  being the images displayed by the system and, respectively, the action taken by the user at the iteration *j*, then the iterative update of the probability estimate of an image  $T_i$  being the target, given the history  $H_t$ , is:

$$P(T = T_i | H_t) = P(T = T_i | D_t, A_t, H_{t-1}) = \frac{P(A_t | T = T_i, D_t, H_{t-1}) P(T = T_i | H_{t-1})}{\sum_{j=1}^n P(A_t | T = T_j, D_t, H_{t-1}) P(T = T_j | H_{t-1})}$$

In computing the probability of a user to take a certain action  $A_t$  given the history so far and the fact that the target is indeed  $T_i$ , namely  $P(A_t|T = T_i, D_t, H_{t-1})$ , a few models were tested. One approach is to estimate the probability of the user to pick an image  $X_a$  from  $X_1, \ldots, X_{n_t}$  by

$$p_{softmin}(A = a | X_1, \dots, X_{n_t}, T) = \frac{exp(-d(X_a, T)/\sigma)}{\sum_{i=1}^{n_t} exp(-d(X_i, T)/\sigma)}$$

and, in the case of choosing any number of images, to assume that each image is selected independently acording to a  $p_{softmin}$ .

**Result presentation** While older versions were displaying the  $n_t$  images with the highest probability of being the target, in the newer version the images selected for display are determined by minimizing the expected number of future iterations estimated by entropy.

**Applications** The system was tested with a database gathering images from 45 CD's of Corel stock photographs.

#### 3.40 PicSOM

**Developer** Laboratory of Computer and Information Sciences, Helsinki University of Technology, Finland.

URL http://www.cis.hut.fi/picsom. A demo of the system is available there too.

#### References [42].

**Features** The following features are computed for a center region, and four regions along the sides. The average R, G, B values give a 15-dimensional vector. The color moments (mean, variance, skewness) of each color channel give a 45-dimensional vector. The Y-values of the YIQ color representation of every pixel's 8-neighborhood are examined, and the estimated probabilities for each neighbor being brighter than



Figure 24: PicSOM querying with relevance feedback.

the center pixel give a 40 dimensional texture feature vector. After applying the Sobel edge detector, each edge pixel direction is put in a 8-bins histogram, which gives a 40-dimensional vector.

Also, the image was normalized  $512 \times 512$  pixels, and the binarized edge image was transformed with the Fourier transform. The magnitude image of the Fourier spectrum was low-pass filtered and decimated by a factor 32, resulting in a 128-dimensional vector.

**Querying** Querying is an iterative process with relevance feedback. The query begins with a fixed selection of representative images. Query improvement is achieved by relevance feedback.

**Matching** The matching is based on the Self-Organizing Map (SOM). The SOM consist of a regular grid of units, each having a feature vector. The SOM is fed with all database images by a sequential regression process, which makes that images are ordered on the grid so that similar images are close to each other. To speed up processing, a multi-level SOM is used to represent the image database. A unit in a higher level SOM represents a cluster of similar images. The distance between the units in the SOM is the Euclidean distance between there feature vectors.

Indexing The indexing datastructure is in fact the array of the SOM.

**Relevance feedback** Each SOM level is treated as a two-dimensional matrix formed of values describing the user's responses to the contents of the seen map units. The matrices are low-pass filtered to spread the responses to the neighboring map units.

**Result presentation** Starting from the SOM unit having the largest positive convolved response, PicSOM retrieves the representative images associated with the map units. If that image has not already been shown to the user, it is presented in the next round.

## 3.41 PicToSeek

Developer Department of Computer Science, University of Amsterdam, The Netherlands.

URL http://www.science.uva.nl/research/isis/pictoseek/. A demo of the system is available at http://zomax.wins.uva.nl:5345/ret\_user/.

#### References [26].

**Features** Color and shape invariants are defined to be used as features in content based queries independent of camera viewpoint, geometry of the objects and illumination conditions. Under the assumption of a dichromatic reflection model, the effect of various imaging conditions on the different color features is analyzed. (In this model, reflection is used to determine how color clusters are formed in the RGB color space by pixels coming from the same uniformly colored surface of the object.) Assuming



Figure 25: PicToSeek result, query at left.

a white illumination and a matte object surface, a basic set of irreducible color invariants is formed by  $\{R/G, B/R, G/B\}$ . Also, it is proven that the normalized color model rgb and a newly defined model  $c_4c_5c_6$ , with  $c_4 = (R - G)/(R + B)$ ,  $c_5 = (R - B)/(R + B)$  and  $c_6 = (G - B)/(G + B)$ , are independent of the viewpoint, illumination direction, illumination intensity and surface orientation of matte objects. The set of color invariants, independent of viewpoint, used in the case of a shiny surfaced object, are  $l_4 = |R - G|/(|R - G| + |B - R| + |G - B|)$ ,  $l_5 = |R - B|/(|R - G| + |B - R| + |G - B|)$ ,  $l_6 = |G - B|/(|R - G| + |B - R| + |G - B|)$ . Based on the above color invariants, normalized color histograms are computed,  $H_A$  defined on the  $c_4$ ,  $c_5$  and  $c_6$  axes and  $H_B$  on  $l_4$ ,  $l_5$ , and  $l_6$  axes. Color invariant gradients are defined using a multi-band approach and a histogram  $H_C$  of  $l_4 l_5 l_6$  color edges is computed. That is because the color gradient for  $l_4 l_5 l_6$  is the one that measures the presence of material edges (not shadow or highlight edges) in an image.

The shape invariants used are the angle between two color edges (invariant under rigid motion plus scaling) and the cross-ratio (projective invariant) between four color edges originating in the same point. A 1D histogram  $H_D$  is constructed on the discretized angle values, and a similar one  $H_E$  on the cross ratio values. Color and shape invariants are also combined in a 4D histogram  $H_F$  such that  $H_F(i, j, k, l)$  counts the number of color edge maxima points of (i, j, k) color and generating an angle l.

**Querying** The query image can be selected from the database collected by the system's Web crawlers or brought in by the user by giving a URL address, see figure 25. Before submitting the query, the user has to select the desired invariance, which indicates to the system how to choose the invariants in computing the similarity. Queries by user-specified feature values are also allowed.

**Matching** The similarity measure between a query histogram  $H_j^Q$  and a database image histogram  $H_j^I$  is the normalized intersection  $d(H_j^Q, H_j^I) = (\sum_{k=1}^n min\{H_j^Q(k), H_j^I(k)\})/(\sum_{k=1}^n H_j^Q(k))$ , with  $j \in \{A, B, C, D, E, F\}$  and n is the number of nonzero bins in  $H_j^Q$ .

Result presentation The retrieved images are shown without explicit order.

#### 3.42 QBIC

(Query By Image Content)

Developer IBM Almaden Research Center, San Jose, CA.

URL http://wwwqbic.almaden.ibm.com/. A demo version of the system is available at site http://wwwqbic.almaden.ibm.com/cgi-bin/stamps-demo. References [60].



Figure 26: QBIC.

**Features** Color features computed are: the 3D average color vector of an object or the whole image in RGB, YIQ, Lab, and Munsell color space and a 256-dimensional RGB color histogram. If x is an n-dimensional color histogram and  $C = [c_1c_2 \dots c_n]$  is a  $3 \times n$  matrix whose columns represent the RGB values of the n quantized colors, the average color vector  $x_{avg}$  is C x. The texture features used in QBIC are modified versions of the coarseness, contrast, and directionality features proposed by Tamura [81].

The shape features consist of shape area, circularity, eccentricity, major axis orientation and a set of algebraic moment invariants. The major axis orientation and the eccentricity are computed from the second order covariance matrix of the boundary pixels: the major axis orientation as the direction of the largest eigenvector and eccentricity as the ratio of the smallest eigenvalue to the largest one. For the database images, these shape features are extracted for all the object contours, semiautomatically computed in the database population step. In this process, the user enters an approximate object outline, which is automatically aligned with the nearby image edges, using the active contours technique. In this object identification step, the user can also associate text to the outlined objects.

The 18 algebraic moment invariants are the eigenvalues of the matrices  $M_{[2,2]}$ ,  $M_{[2,3]} \times M_{[3,2]}$ ,  $M_{[3,3]}$ ,  $M_{[3,4]} \times M_{[4,3]}$ ,  $M_{[4,4]}$ ,  $M_{[4,5]} \times M_{[5,4]}$ , where the elements of  $M_{[i,j]}$  are scaled factors of the central moments.

QBIC also implemented a method of retrieving images based on a rough user sketch. For this purpose, images in the database are represented by a reduced binary map of edge points. This is obtained as follows: first, the color image is converted to a single band luminance; using a Canny edge detector, the binary edge image is computed and is next reduced to size  $64 \times 64$ . Finally this reduced image is thinned.

**Querying** QBIC allows queries based on example images, user-constructed sketches or/and selected color and texture patterns. In the last case, the user chooses colors or textures from a sampler. The percentage of a desired color in an image is adjusted by moving sliders.

**Matching** For the average color, the distance between a query object and database object is a weighted Euclidean distance, where the weights are the inverse standard deviation for each component over the samples in the database. In matching two color histograms, two distance measures are used: one low-dimensional, easy to compute (the average color distance) and one much more computationally expensive (the quadratic histogram distance). The first one (which is computed for all the images in the database) acts as a filter, limiting the expensive matching computation to the small set of images retrieved by the first matching. The average color distance is  $d_{avg}^2(x, y) = (x_{avg} - y_{avg})^t (x_{avg} - y_{avg})$ . The histogram

quadratic distance is given by  $d_{hist}^2(x,y) = (x-y)^t A(x-y)$ , where the symmetric color similarity matrix A is given by  $a_{ij} = 1 - d_{ij}/d_{max}$ , with  $d_{ij}$  being the  $L_2$  distance between the colors i and j in the RGB space and  $d_{max} = max_{i,j}d_{ij}$ . The texture distance is a weighted Euclidean distance, with the weighting factors being the inverse variances for each of the three texture components over the entire database. Two shapes are matched also by a similar weighted Euclidean distance between shape feature vectors.

In a query by sketch, after reducing the binary sketch image drawn by the user to size  $64 \times 64$ , a correlation based matching is performed, a kind of template matching. This is done by partitioning the user sketch into  $8 \times 8$  blocks of  $8 \times 8$  pixels and find the maximum correlation of each block of the sketch within a search area of  $16 \times 16$  pixels in the image database (this is done by shifting the  $8 \times 8$  block in the search area). This local correlation score is computed on the pixel level using logical operations.

The matching score of a database image is the sum of the correlation scores of all local blocks. **Indexing** QBIC was one of the first systems that applied multidimensional indexing to enhance the speed performance of the system. The average color and the texture features (both 3D vectors) are indexed using  $R^*$ -trees. The 18-dimensional moment-based shape feature vector is first reduced using the KL transform and then indexed by using  $R^*$ -trees.

**Result presentation** The best matches are presented in decreasing similarity order with (optionally) the matching score aside.

**Relevance feedback** Any retrieved image can be used as a seed for a subsequent query by example.

**Applications** At http://wwwqbic.almaden.ibm.com/tmdemo/ is a demonstration of the QBIC system as trademark server.

## 3.43 Quicklook<sup>2</sup>

Developer CNR Institute of Multimedia Information Technologies, Milan, Italy.

URL http://www.itim.mi.cnr.it/Linee/Linea1/Sottolinea3/relfeme.htm. References [18].

**Features** Besides text descriptors, Quicklook<sup>2</sup> uses the following contest-based features. The color histogram of 64 bins. The color coherence vector in Lab color space, quantized into 64 colors, where pixels are coherent if they belong to a large similarly colored region; the image is first blurred by averaging a  $3 \times 3$  neighborhood. The percentage of skin-colored pixels. The 'spatial chromoticity histogram': bin *i* 



Figure 27: Quicklook<sup>2</sup>.

gives the ratio h(i) of pixels having color *i*, the relative coordinates of the baricenter  $\mathbf{b}(\mathbf{i})$  of these pixels, and the standard deviation  $\sigma(i)$  of these pixels' distance to the baricenter. The cooccurrence of colors in Lab color space, quantized into 11 colors: red, orange, yellow, green, blue, purple, pink, brown, black, gray, white. The moments of inertia (mean, variance, skewness, kurtosis) of the color distribution in Lab space. Contrast information at Canny edge pixels: the percentage of low, medium, and high contrast pixels, the thresholds on the gradient strengths corresponding to medium and high constrast, the number of connected regions with high contrast contours, and the percentage of medium contrast edge pixels connected to high contrast edges. Texture features based on the 'neighborhood gray tone difference matrix' [3]: coarseness, contrast, busyness, complexity, and strength. The mean and variance of the absolute values of the Daubechies wavelet transform coefficients of the luminance of the subimages at the first three levels of resolution. The histrogram of 72 bins of directions of high gradient pixels on Canny edges. The Hu moment invariants [33]. After quantization into 11 colors: the number of color regions, the distribution of these regions around the center, x, and y axis.

**Querying** The user initially selects a query image. The features of this image are taken for the first search. The user can than indicate which images are relevant. From these relevant image a new query feature vector is composed, see below.

**Matching** The distance between two spatial chromoticity histograms  $h_1, h_2$  of lenght k is defined as  $\sum_{i=1}^{k} \min\{h_1(i), h_2(i)\}(1 - d(\mathbf{b_1}(\mathbf{i}), \mathbf{b_1}(\mathbf{i}))\sqrt{2}/2) + \min\{\sigma_1(\mathbf{i}), \sigma_2(\mathbf{i})\}/\max\{\sigma_1(\mathbf{i}), \sigma_2(\mathbf{i})\}$ . The distance measure for any of the other features is the  $L_1$  distance. Each distances measure  $d_j$  is normalised by a factor  $\mu_j + K\sigma_j$ , with  $\mu_j$  and  $\sigma_j$  the average and the standard deviation over all images, and K, which affects the number of outliers, set to 3. The total distance between two images is a weighted sum of the feature distance measures  $d_j$ .

**Indexing** In a preprocessing step, the distance of all database objects to m reference objects are computed. At query time, the distances  $d_i$ , i = 1, ..., m of the query to these reference objects is also computed. With an SQL query, those database objects are retrieved that lie within some range from each  $d_i$ . After this filtering, these database objects are compared to the query directly.

**Result presentation** The results are listed in descreasing order of similarity. See figure 27 for the result after a few iterations of relevance feedback.

**Relevance feedback** The innovative part of the system is the relevance feedback method. After the relevant images are selected, each of their features contributes to the new query feature vector if its distance to the average over the relevant images' feature is sufficiently large (three times the standard deviation). The new query feature vector is the average of the contributing features.

#### **3.44 RETIN**

(Recherche et Traque Interactive)

Developer ENSEA/University of Cergy-Pontoise, France.

URL http://http://www-etis.ensea.fr/~image/Indexation.html/. References [24].

**Features** From each data base image, rondom pixels are selected. Of these pixels the RGB, rgb, HSV, Lab, and Luv values are taken. The texture feature of these pixels consist of the energie response of 12 Gabor filers (four different direction and three different frequencies). All these values over all data base images are clustered using a Kohonen self organizing neural network. Then, for each individual image, the selected pixels are classified, and the set of fequencies of each cluster, a 1D histogram whose bins are the classes, constitutes the image signature.

**Querying** The user give a query image. Searching is either done on the whole image, or on a region of interest that is drawn by the user. From the whole image or the region of interest, the distribution of feature classes is computed, and compared with those of the database images.

**Matching** Two distances between two histogram vectors  $H_1, H_2$  of lenght k are used. One is the weighted Minkowski distance of order p:  $d_1(H_1, H_2) = (\sum_{i=1}^k w_i |H_1[i] - H_2[i]|^p)^1/p$ , with the weights summing to one. The other allows cross matching between bins:  $d_2(H_1, H_2) = (\sum_{i=1}^k \sum_{j=1}^k w_{ij} |H_1[i] - H_2[i]|^p)^1/p$ , with the weights summing to one. These distances are computed for all the different color and



Figure 28: Shoebox querying with a region.

texture features, which are averaged with weights  $u_i$ .

**Result presentation** The retrieved images are shown in linear order with respect to the matching score. **Relevance feedback** The weights between the feature distances  $u_i$  and within the distances  $w_i$  and  $w_{ij}$ , are set uniformly to start with. The user indicates which of the retrieved images are relevant, and which are not. The system then adjusts the weights using a least mean square error rule.

## 3.45 Shoebox

**Developer** AT&T Laboratories, Cambridge, UK. The development was started at Olivetti and Oracle Research Laboratory.

URL http://www.uk.research.att.com/dart/shoebox/. A video about the system is at http://ftp.uk.research.att.com/pub/videos/qsif-200/shoebox-qsif-200.mpg. References [55].

**Features** The images are annotated by keywords using speech recognition. The automatic features extracted are the average color in HSV color space, and the variances in each of the color channels of regions. The regions are either resulting from color segmentation, or a fixed partitioning of the image, or the overlay of both.

**Querying** Querying can be done with the spoken annotations. Another way is to select a region from an image in the database to find similar images.

**Matching** The Euclidean distance between feature vectors for use with the M-tree. For use with a B-tree, the feature space is quantised, and the corresponding values are usd in an inverted index.

Indexing Both the B-tree and the M-tree are used for indexing.

**Result presentation** When querying with regions, the system can highlight those regions in the resulting images that match the query region.

Applications The system is meant as a tool to manage personal photo collections.

Performance The M-tree indexing outperforms the B-tree indexing.

## **3.46 SIMBA**

(Search Images By Appearance)



Figure 29: SIMBA query result.

**Developer** Institute for Pattern Recognition and Image Processing, Freiburg University, Germany.

URL http://lmb.informatik.uni-freiburg.de/SIMBA. A demo is available at web address http://obelix.informatik.uni-freiburg.de/cgi-bin/SIMBA/SIMBA.cgi. References [72].

**Features** The system uses a histogram of a function f of transformed local pixel patterns of image I(x, y), taken over all rigid motions g. For example, f(I) could be  $I(0, 1) \cdot I(2, 0)$ , and the histograms could be constructed of the values  $g_{t_0,t_1,\phi}(I(0,1)) \cdot g_{t_0,t_1,\phi}(I(2,0)), t_0 = 0, \dots, n, t_1 = 0, \dots, m, \phi = 0, P - 12\pi/P.$ This is based on the observation that the integration over all transformations g (Haar integral) of f(q(I))makes f invariant under these transformations. The histogram is used rather than the integral, to keep more local information. By combining different functions f and different color layers (RGB in the online demo, however, also HSV, luminance and chromaticities are supported) a multidimensional histogram is built. A fuzzy version of the histogram was developed to get rid of the discontinuities of the histogram operator at bin boundaries. The evaluation of the features is speeded up by Monte Carlo estimation instead of a full calculation.

**Querying** The user is presented with a selectable number (1-25) of random images from the database. By selecting one of these, a query for similar images is started. Alternatively, the user can specify a URL for his own query image, whose size must be comparable to the size of the database images (384x256).

**Matching** The system provides histogram intersection (used in the demo), as well as  $\chi^2$ ,  $L_1$ , and  $L_2$ distances for matching.

Indexing No indexing data structure is used.

**Result presentation** The results of a query are shown in decreasing order of match row by row. The user may specify resolution and number of rows/columns for the results, before starting a query. The result display can be used for a new query by clicking on one of the displayed images (to come iteratively closer to the desired image type) or by specifying a new URL, see figure 29.

#### 3.47 **SMURF**

(Similarity-based Multimedia Retrieval Framework)

Developer Center for Geometry, Imaging, and Virtual Environments, Institute of Information and Computing Sciences, Utrecht University, The Netherlands.



Figure 30: Hieroglyphic retrieval with SMURF.

#### URL http://give-lab.cs.uu.nl.

#### References [87].

**Features** It is a framework that incorporates color, texture, and shape, but the system is primarily oriented towards shape-based retrieval research. In particular there are demo's for polyline matching, and weighted point set matching.

**Querying** The search is started by selecting an image from a random selection. The user can select a new image from the search results so as to narrow down to the target images.

**Matching** Matching of two polylines is done by computing the area between the two corresponding turning angle functions, and minimizing over rotations (vertical shift of the function), and starting position (horizontal shift). The weighted point set matching is done with a pseudometric, the Proportional Transportation Distance. Unlike the Earth Movers Distance, it obeys the triangle inequality for sets of unequal weight, which is essential for the indexing method used.

**Indexing** The triangle inequality and similarity of similarity measures is used to make the search more efficient. This is based on the following observation. Consider a shape or point set  $A_1$  that closely matches a query  $A_q$ :  $d(A_1, A_q)$  is small. Let  $A_r$  be some reference shape, called vantage object. If the triangle inequality holds,  $d(A_r, A_q) \leq d(A_r, A_1) + d(A_1, A_q)$ , then we know that  $d(A_r, A_q) - d(A_r, A_1)$  is small as well. We can measure the distance between a database shape  $A_1$  and a query  $A_q$  by comparing their distances from a vantage object  $A_r$ . The computation of the latter distances can be done off-line for all  $A_i$ . So at run-time the computation of a single  $d(A_r, A_q)$  yields all  $A_i$  that resemble  $A_r$  as much as  $A_q$ , including those that resemble  $A_q$ . Using multiple vantage objects, say m, increases precision, recall is always 100%. A similar approach was taken in FIDS [8]. In SMURF, the search is done with an optimal approximate k-nearest neighbor search in the m-dimensional 'vantage space' of distances. In this way, the number of expensive similarity measures between shapes is reduced to m. The search further only takes cheap distance computations in vantage space.

**Result presentation** The results are shown in decreasing order of match score.

**Applications** The polyline matching is applied to about 70,000 polylines from a collection of hieroglyphics that is used as a font for a hieroglyphic text processor, see figure 30. The weighted point set matching is used for searching in a collection of 1100 fish contours.

## 3.48 SQUID

(Shape Queries Using Image Databases)



Figure 31: SQUID result, query at top left.

Developer Centre for Vision, Speech, and Signal Processing, University of Surrey, UK.

URL http://www.ee.surrey.ac.uk/Research/VSSP/imagedb/demo.html. A demo is available at http://www.ee.surrey.ac.uk/Research/VSSP/imagedb/squid.html.

#### **References** [56].

**Features** The boundary contour extracted from an image is represented by three global shape parameters (eccentricity, circularity and aspect ratio of the Curvature Scale Space image) and a descriptor obtained from the CSS image. The CSS image of a closed planar curve,  $\Gamma$ , is computed by repeatedly convolving its natural parameterization,  $\{(x(u), y(u)) \mid u \in [0, 1]\}$ , with a Gaussian function of increasing standard deviation  $\sigma$  and representing the curvature zero-crossing points (inflection points) of the resulting curve,  $\Gamma_{\sigma} = \{(x(u) * g(u, \sigma), y(u) * g(u, \sigma)) \mid u \in [0, 1]\},$  in the  $(u, \sigma)$  plane. The shape descriptor is made of the maxima of the CSS image contours.

Querying The user selects one of the displayed object boundaries in order to generate a query for similar contours (see figure 31).

**Matching** First, from the database are chosen those models whose global parameters are sufficiently close to the query global parameters. To these images a CSS matching algorithm is applied. This consists of first finding a circular shift of one of the two sets of maxima so that they best match the other set and then computing the matching score as the sum of the Euclidean distances between the matched pairs plus the vertical coordinates of the unmatched maxima.

**Result presentation** The system displays the first *n* matched images in decreasing similarity order. Under each retrieved image, an integer match value is shown.

**Applications** The system database consists of 1100 images of marine creatures on a uniform background.

#### 3.49 Surfimage

Developer INRIA, Rocquencourt, France.

URL http://www-rocg.inria.fr/imedia/index UK.html. A demo of the system is available at http://www-rocq.inria.fr/cqi-bin/imedia/surfimage.cqi.

## References [59].

Features Surfimage gives the possibility of combining various low-level and high-level features. The lowlevel features are RGB color histogram, edge-orientation histogram computed after applying a Canny edge



Figure 32: SurfImage.

detector, texture signature derived from the gray-level cooccurrence matrix, Fourier transform, and Wavelet transform. High-level features include eigenimages (see Photobook), deformable intensity surfaces, and intensity surface curvature histogram. The latter is a normalized histogram of

$$S_I(p) = \frac{1}{2} - \frac{1}{\pi} \arctan \frac{k_1(p) + k_2(p)}{k_1(p) - k_2(p)}$$

at each point p = (x, y, I(x, y)) over the entire image, where  $k_1, k_2$  are the prinicpal curvature values. **Querying** The user selects image features and appropriate similarity measures from the lists displayed and specifies weights to define the importance of each feature in the retrieval. He also can select a database (among the databases available are the MIT face database used in Photobook, the MIT Vistex database of textures, a paintings database, the BTphoto database of city/country scenes and a combined database). **Matching** Various similarity metrics are implemented in Surfimage. Among them are standard metrics,

such as the  $L_p$  distances (most often used are  $L_1$  and  $L_2$ ), the cosine metric  $d_{cos}(x, y) = 1 - x \cdot y/||x||||y||$ , or statistical metrics such as the Hellinger metric

$$d_H(x,y) = 1 - \sum_{i=1}^n \sqrt{\frac{x_i y_i}{|x||y|}},$$

where ||x|| is the  $L_2$  norm and |x| is the  $L_1$  norm of the vector x. For histograms the intersection metric is used.

In the case of choosing more than one feature for the query, the corresponding similarity measures are combined in a final weighted measure. Before this, some of the distance measures are normalized so that they have values in [0, 1]. If  $\mu_i$  and  $\sigma_i$  are the mean and standard deviation of the distance d for the feature i, this distance is normalized to:  $d_n(x^i, y^i) = [d(x^i, y^i) - (\mu_i - 3\sigma_i)]/6\sigma_i$ .

**Relevance feedback** Surfimage has a relevance feedback mechanism, which takes into consideration both relevant and nonrelevant examples from the user, in a statistical approach that estimates the distribution of relevant images.

#### 3.50 SYNAPSE

(SYNtactic APpearance Search Engine)



Figure 33: Synapse query with the drawn region of interest.

**Developer** Center for Intelligent Information Retrieval, University of Massachusetts, Amherst, MA. URL http://hobart.cs.umass.edu/~mmedia/. A demo of the system is available at the address http://cowarie.cs.umass.edu/~demo/Demo.html.

#### References [53].

**Features** Queries are made on the visual appearance of the objects in the image, defined as the shape of the intensity surface and represented using the outputs of a set of Gaussian derivative filters. More precisely, each image is filtered at uniformly sampled points with Gaussian derivative up to the order 2. Thus, for each sampled point p = (x, y) of intensity I(p), the so-called 2-jet at this point is given by the vector  $J^2[I](p,\sigma) = \{I_{\sigma}(p), I_{x,\sigma}(p), I_{y,\sigma}(p), I_{x^2,\sigma}(p), I_{xy,\sigma}(p), I_{y^2,\sigma}(p)\}$ , where  $I_{x^iy^i,\sigma}(p)$  is the image convoluted with the (i, j)-th partial derivative of the Gaussian function of scale  $\sigma$ :  $I_{x^iy^i,\sigma}(p) = I(p) * \sigma^{i+j}\partial^{i+j}G(p,\sigma)/\partial x^i\partial y^i$ ,  $i, j \in \{0,1,2\}$ . From this vector, four differential invariants are computed:  $d_1^{\sigma} = I_x^2 + I_y^2$ ,  $d_2^{\sigma} = I_{x^2} + I_{y^2}$ ,  $d_3^{\sigma} = I_{x^2}I_x^2 + 2I_{xy}I_xI_y + I_{y^2}I_y^2$ ,  $d_4^{\sigma} = I_{x^2}^2 + 2I_{xy}^2 + I_{y^2}^2$ . Computing these four invariants at three different scales,  $\sigma_1, \sigma_2, \sigma_3$ , a multi-scale invariant vector D is generated for each sampled point p. Along with each multi-scale invariant vector D, the coordinates (x, y) of the corresponding sampled point and the number i of the image are also stored.

Querying Querying is done by example (see figure 33 where a region has been marked).

**Matching** Matching is done in two phases. First, for every invariant vector  $D_n$  of the query image, a number of invariant vectors stored in the database are found so that they lie in the range  $[D_n - t, D_n + t]$ . This is done by finding for each component  $D_n(j)$  a list of invariant vectors, whose *j*-component lies in the interval  $[D_n(j) - t(j), D_n(j) + t(j)]$  and intersecting the lists.

In the second phase, a spatial check is performed on the list of candidate images. For each sampled point  $q_m$  of the query image, the number  $S_m$  of query points whose Euclidean distance to the  $q_m$  is within a threshold factor of the Euclidean distance of their corresponding points in the candidate image. The score of the candidate image is given by max<sub>m</sub>  $S_m$ .

**Indexing** An index is created on each of the 12 coordinates of the invariant vector D. Instead of storing each invariant vector in one record of the form (i, x, y, D), 12 smaller tables (inverted file lists), each with 4 columns (D(p), i, x, y), are generated. Each inverted file list is stored in a binary tree sorted on the first column.

**Result presentation** Images are showed in decreasing similarity score.



Figure 34: TODAI: query specification.

**Relevance feedback** The search can be refined by marking the relevant images found among the results of a query and reiterate the query.

## **3.51 TODAI**

(Typographic Ornament Database And Identification)

**Developer** Electrical Engineering Department, EPFL, Switzerland, and later the Intelligent Systems Laboratory, Halmstad University, Sweden.

#### URL

http://www.unil.ch/BCU/docs/collecti/res\_prec/en/wtres/todai\_doc.html.
References [54].

**Features** The image is first decomposed into six orientation images. Each orientation image is result of applying a filter that passes edges with a specified orientation. Thus, an orientation image contains information about the presence in the original image of linear structure in the direction of the pass orientation. The projection of the orientation image in the direction of the pass orientation is called an orientation radiogram. Six fixed directions are chosen to make radiograms. Each radiogram is represented by a feature vector of the real parts and the absolute values of the imaginary parts of the first ten Fourier coefficients. The absolute value operation provides invariance to flipping of the image.

**Querying** The user can specify the location of a query image. Additional criteria such as the size, use and nature of the ornament can be indicated, see figure 34. There is a possibility to make an advanced search combining keywords and the image content.

**Matching** The similarity between two images is the sum of the Euclidean distances of the six feature vectors.

Result presentation The best match is shown, the user can browse through the list of next matches.

**Applications** The system is used in Passe-Partout, the International Bank of Printers' Ornaments (see http://www.unil.ch/BCU/docs/collecti/res\_prec/fr/todai\_intro.html), which stores 2316 images.

## 3.52 Viper

(Visual Information Processing for Enhanced Retrieval)

Developer Centre Universitaire d'Informatique, University of Genova, Switzerland.

URL http://viper.unige.ch/. References [77]. Features Querying and Result presentation Matching Relevance feedback Indexing Applications

## 3.53 VIR Image Engine

The VIR Image Engine is an extensible framework for building content based image retrieval systems. **Developer** Virage Inc.

URL http://www.virage.com/products/vir-irw.html.

#### References [4].

**Features** A basic concept is that of a *primitive*, which denotes a feature's type, computation and matching distance. Five abstract data types are defined: global values and histograms, local values and histograms, and graphs. The VIR Image Engine provides a set of general primitives, such as global color, local color, texture and shapes. Appart from these, various domain specific primitives can be created when developing an application. When defining such a primitive, the developer supplies a function for computing the primitive's feature data from the raw image.

**Querying and Result presentation** The VIR Image Engine provides a set of GUI tools necessary for the development of a user interface. These include facilities for image insertion, image query, weight adjustment for re-query, inclusion of keywords, and support for several popular image file formats. Another available component, the query canvas, allows queries-by-sketch; it consists of a bitmap editor where the user can sketch a picture with drawing tools and color it using the colors from a palette. Also, the user can bring onto the canvas an image from an existing collection and modify it using the same drawing tools. Queries can be performed on various user-defined combinations of primitives.

**Matching** When defining a new primitive, a function for computing the similarity between two sets of feature data previously extracted must also be supplied by the developer. When comparing two images, for each primitive in the current query combination, a similarity score is computed using the distance function defined within the primitive. These individual scores are combined in an overall score using a set of weights in a way characteristic to the application. This score is then stored in a *score structure*, which contains also the individual similarity scores for each primitive. This allows a quick recomputation of the overall score for a new set of weights.

**Relevance feedback** A sort of relevance feedback is obtained by searching for matches to a fixed query image for different sets of weights.

**Indexing** The storage of feature data for all primitives is the responsibility of the application developer.

**Applications** The system was integrated into databases from Sybase, Object Design, and Objectivity, and added as a component to the Oracle DBMS. The AltaVista Photofinder and Illustra's Visual Intelligence system are two applications of Virage technology.

## 3.54 VisualSEEk

Developer Image and Advanced Television Lab, Columbia University, NY.

URL http://www.ctr.columbia.edu/VisualSEEk/.

#### References [76].

**Features** In the database population step, each image is automatically decomposed into regions of equally dominant colors. For each region, feature properties and spatial properties are retained for the subsequent queries. A query consists of finding the images that contain the most similar arrangements of similar regions.



Figure 35: VisualSEEk query interface.

The color region extraction uses the back-projection technique. The first step is the selection of a color set. This is a 166-dimensional binary vector c, which defines a selection of 166 colors in the HSV color space. From a given image I, another image B is generated by  $B[x, y] = \max_j a[k, j]c[j]$ , where  $k \in \{0, \ldots, 165\}$  is the index of the color of the pixel (x, y) in the image I and a[k, j] is the similarity between two colors (with the indices k and j) in the HSV space. Next the image B is filtered and spatially localized color regions are extracted.

Along with the color set used for back-projection, the region centroid, area (defined as the number of pixels in the region) and the width and height of the minimum bounding rectangle are also stored.

**Querying** To start a query, the user sketches a number of regions, positions and dimensions them on the grid (see figure 35) and selects a color for each region. Also, the user can indicate boundaries for location and size and/or spatial relationships between regions. After the system returns the thumbnail images of the best matches, the user is allowed to search by example using the returned images.

**Matching** To find the matches of a query image with a single region, queries on color set, region absolute location, area and spatial extent are first done independently. The color set similarity is computed by  $d(c_q, c_t) = (c_q - c_t)^t A(c_q - c_t)$ , where  $c_q, c_t$  are two color sets and A = (a[i, j]) is the color similarity matrix. If the user has defined spatial boundaries for the query region, then its distance to a target region is 0 if the target region centroid falls inside this boundaries, and is given by the Euclidean distance between the centroids otherwise. The distance in area between two regions is the absolute value of the difference, while the distance between the minimum bounding rectangles,  $(w_q, h_q)$  and  $(w_t, h_t)$ , of two regions is the  $L_2$  metric. The results of these queries are intersected and from the obtained candidate set, the best matching images are taken by minimizing a total distance given by the weighted sum of the four distances mentioned.

If the query image consists of a number of regions, in absolute or relative location, then for each region positioned in absolute location, a query like that described above is made, and for regions positioned by relative location individual queries on all attributes except location are performed. For the intersection of all this query results, the relative spatial relations specified by the user are evaluated using 2D string representation [16].

**Indexing** The color set distance can be written as  $d(c_q, c_t) = \mu_q + \mu_t - 2c_q^t r_t$ , where  $\mu_q = c_q^t A c_q$ ,  $\mu_t = c_t^t A c_t$  and  $r_t = A c_t$ . For each target region,  $\mu_t$  and  $r_t[m]$ ,  $m \in \{0, \dots, 165\}$ , are indexed individually. The centroids of the image regions are indexed using a quadtree. For the indexing of the minimum bounding rectangles, R-tree are used.



Figure 36: SaFe query interface and result.

**Result presentation** The results of a query are displayed in decreasing similarity order. Under each retrieved image, the distance to the query image is indicated.

**Performance** Another system, SaFe [74], see figure 36, was developed in order to improve the performances of VisualSEEk. SaFe was suppose to extend the query capabilities to other features, such as texture, and to incorporate more powerful spatial indexing techniques. There is no indication that this has been done. The two systems seems to differ only in the user interface.

## 3.55 VP Image Retrieval System

**Developer** National Center for Science Information Systems, University of Tokyo, Japan. URL http://www.rd.nacsis.ac.jp/.

References [37].



Figure 37: VP.

**Features** The silhouettes extracted from a database image are first decomposed into convex parts. The decomposition process is recursively performed as follows: at each negative curvature point  $V_i$ , an erosional vector, directed to the interior of the shape contour and proportional to the global angle and length of the concavity at  $V_i$ , is computed. Based on this vector, the speed to erode away the current part of the contour from a negative curvature point  $V_i$  to a point  $V_j$  on the other side of the contour is defined. The shape contour will be split by a line segment  $V_i V_j$  of minimum erosion time.

Next, for each part of the decomposition a number of features are extracted: *attributes* of the convex part (horizontal/vertical position, scale, shape, skeleton length, curvature and slope) and *relations* between two adjacent parts (relative horizontal/vertical position, relative scale, position of the skeletons intersection and angle between the two skeletons). In order to represent these features, their value domain is quantized into a small number of levels (e.g., 3 for the length attribute, which correspond to small, medium, large), each feature being associated a fixed number of bits in a binary sequence. Two binary vectors are constructed. The first one is the concatenation of all attribute sequences of the parts in the decomposition (such a sequence is called primitive signature). The second one, retains the relations between the parts along with their attributes.

**Querying** Queries are formulated in three steps. First, a silhouette is drawn, this is next automatically decomposed into corresponding convex parts, and finally the user associates query weights to these parts. These weights specify the degree of relevance in the query process, from not important to very important, of the computed attributes and relations of the query parts. When the user chooses to relax some of the attributes of particular parts, the system represents this by adding extra 1 bit values to the primitive signatures of these parts in the sequences corresponding to the selected attributes.

**Matching** The matching is done by shifting each query primitive signature over the concatenated primitive signatures of the target image and counting the matches  $C_a$ . Similar shifting and conjunction operations are applied to the second vector, storing the relations between parts, with the number of matched relations computed being  $C_r$ . The similarity score is given by  $S = (C_a + C_r)/(N_a + N_r)$ , where  $N_a$  is the total number of primitive signatures of the first query vector, and  $N_r$  is the total number of relations in the other query vector.

#### 3.56 WebSEEk

Developer Image and Advanced Television Lab, Columbia University, NY.

URL http://www.ctr.columbia.edu/WebSEEk/.

#### References [74].

**Features** WebSEEk makes text-based and color based queries through a catalogue of images and videos collected from the Web. Color is represented by means of a normalized 166-bin histogram in the HSV color space, see VisualSEEk.

**Querying** The user initiates a query by choosing a subject from the available catalogue or entering a topic. The results of the query may be used for a color query in the whole catalogue or for sorting the result list by decreasing color similarity to the selected item. Also, the user has the possibility of manually modifying an image/video color histogram before reiterating the search.

**Matching** The distance between a query color histogram  $h_q$  and a target histogram  $h_t$  is given by  $d(h_q, h_t) = (h_q - h_t)^t A(h_q - h_t) \approx \mu_q + \mu_t - 2 \sum_{k \text{ with } h_q[k] \ge \tau} h_q[k] r_t[k]$ , where A = (a[i, j]) is a color similarity matrix,  $\mu_q = h_q^t Ah_q$ ,  $\mu_t = h_t^t Ah_t$ ,  $r_t = Ah_t$  and  $\tau$  defines a threshold for choosing the most significant colors in the approximated distance.

**Indexing** For all the database images,  $\mu_t$  and  $r_t[k]$ ,  $k \in \{0, \dots, 165\}$  are computed and indexed individually.

**Result presentation** Images retrieved are displayed page by page in a descending similarity order. A manipulation (union, intersection, subtraction) of the search result lists, in order to reiterate a query, is possible.

**Relevance feedback** The user has the possibility of selecting positive and negative examples from the result of a query in order to reformulate the query, see figure 38. If the set of relevant images is denoted by  $I_r$  and the set of nonrelevant images is  $I_n$  then the new query histogram at the (k + 1)th feedback iteration



Figure 38: WebSEEk relevance feedback.

is computed by  $h_q^{k+1} = \|\alpha h_q^k + \beta \sum_{i \in I_r} h_i - \gamma \sum_{j \in I_n} h_j\|.$ 

#### 3.57 WebSeer

Developer Department of Computer Science, University of Chicago, Illinois, USA.

**URL** http://infolab.cs.uchicago.edu/webseer/.

#### References [80].

**Features** The images collected from the Web are submitted to a number of color tests in order to separate photographs from drawings. Some simple tests measure the number of different colors in the image, the fraction of pixels which have one of the N most frequent colors for a given threshold N, the fraction of pixels with transition value (the largest  $L_1$  distance between a pixel and one of its neighbors in the RGB space) greater than a threshold T, the fraction of pixels with saturation level (the difference between the maximum and the minimum value of the RGB color bands) greater than a threshold T, and the ratio of the image dimensions. A more elaborate test creates first an average color histogram for graphics,  $H_g$ , and one for photographs,  $H_p$ , using two large sets of images. Defining the correlation between two normalized RGB histograms, A and B, as  $C(A, B) = \sum_{i=0}^{15} \sum_{j=0}^{15} \sum_{k=0}^{15} A_{i,j,k} B_{i,j,k}$ , an image with a color histogram  $H_i$  gains a score to the test equal to  $s = C(H_i, H_p)/(C(H_i, H_p) + C(H_i, H_g))$ . Two similar tests are also made using the farthest neighbour histograms and the saturation histograms instead of the color histograms.

Images determined to be photographs are subjected to a face detector based on a neural network. Keywords are extracted from the image file name, captions, hyperlinks, alternate text and HTML titles.

**Querying** The user gives keywords describing the contents of the desired images, and optionally specifies some image characteristics such as dimensions, file size or whether he is looking for photographs or drawings. In the case the user is looking for people, he must indicate the number of faces as well as the size of the portrait.

**Matching** To classify an image as photograph or graphics, the color tests are combined using multiple decision trees constructed using a training set of hand-classified images. They are binary trees whose internal nodes contain the next test the image should be submitted to and a threshold to direct the search by comparing the test result with the threshold. In each leaf we find a probability estimate that the image is a photograph. Computing the mean of the results got from all decision trees and comparing this with a threshold, a decision is taken whether the image falls in one category or the other.

**Result presentation** The thumbnails of the resulting images are displayed by decreasing size of the face in the case of searching for portraits. Otherwise, there is no explicit order of the retrieved images. The user has access to the Web page where the image was collected from, through a page icon aside the displayed thumbnail.

#### 3.58 WISE

(Wavelet Image Search Engine)

Developer Department of Computer Science, Stanford Univerity.

URL http://www-db.stanford.edu/~wangz/project/imsearch/. A demo of the system
is available at http://bergman.stanford.edu/~zwang/project/imsearch/WBIIS.html.
References [89].

**Features** The system, also called WBIIS (Wavelet-Based Image Indexing and Searching), performs queries on color layout information encoded using Daubechies wavelet transforms. In a preprocessing step, the image is rescaled to  $128 \times 128$  pixels and converted from RGB color space representation to another color space. This color space, defined by  $C_1 = (R + G + B)/3$ ,  $C_2 = (R + (max - B))/2$ ,  $C_3 = (R + 2 * (max - G) + B)/4$ , with max the maximum possible value for each RGB color component, is selected because of its similarity to the opponent color axes used by the human visual system. On each of the three color components, a 4-layer 2D fast wavelet transform using Daubechies wavelets is applied. From the resulting  $128 \times 128$  matrices, denoted as  $W_{C_1}$ ,  $W_{C_2}$ , and  $W_{C_3}$ , each  $16 \times 16$  upper left corner is stored as part of the feature vector (the  $8 \times 8$  upper left corner, representing the lowest frequency bands in the wavelet transforms, accounts for object configurations in the image, while the three surrounding  $8 \times 8$ submatrices represent detail information). Also, the standard deviation,  $\sigma_{C_i}$ , of the  $8 \times 8$  upper left corner of  $W_{C_i}$  is computed and stored as part of the feature vector.

**Querying** Querying is done by example, where the query image is either a database image or a hand drawn sketch. Also low resolution images or partial images (images that contain non-specified areas represented by black rectangles) can be query images to the system.

**Matching** The matching process is performed in two steps. In the first step, the standard deviations computed for the query image,  $\sigma_{C_i}^q$ , are compared with the corresponding values,  $\sigma_{C_i}^i$ , stored for each database image. If the acceptance criterion,  $(\sigma_{C_1}^q\beta < \sigma_{C_1}^i < \sigma_{C_1}^q/\beta) \lor ((\sigma_{C_2}^q\beta < \sigma_{C_2}^i < \sigma_{C_2}^q/\beta) \land (\sigma_{C_3}^q\beta < \sigma_{C_3}^i < \sigma_{C_3}^q/\beta))$ , with the threshold  $\beta$  usually set to 0.5, is not achieved, the distance between the two images is set to 1. In the second step, the images that passed the first matching phase are compared to the query using a weighted Euclidean distance between wavelet transform coefficients:  $\sum_{m=1}^2 \sum_{n=1}^2 w_{mn} \sum_{i=1}^3 (w_{C_i} d_E(W_{C_i,m,n}^i, W_{C_i,m,n}^q))$ , where  $W_{C_i,1,1}$  and  $W_{C_i,1,2}, W_{C_i,2,1}, W_{C_i,2,2}$  denotes the 8 × 8 upper left corner and, respectively, the three surrounding 8 × 8 submatrices of  $W_{C_i}$ , and



Figure 39: WISE result with top left query.

with  $d_E$  the Euclidean distance. By raising  $w_{C_2}$  or  $w_{C_3}$  the importance of color variation in the image is emphasized, while raising  $w_{2,1}$ ,  $w_{1,2}$  or  $w_{2,2}$  the vertical, horizontal, or diagonal edge details in the image are emphasized.

Result presentation The first few matches are presented in decreasing similarity order.

**Applications** The system is used at Stanford University Library to assist teaching and research projects in liberal art departments.

# **4** Summary

The features used in the 46 systems discussed in this article are listed in table 1. The features are classified into the low level classes color, texture, and shape, and the higher level classes layout and face detection. The use of keywords is also listed.

		Color							Texture								Shape												
System	Reference	Eigen image	Global/subimage histogram	Correllation histogram	Color moments	Color coherence vector	Region histogram	Dominant colors	Other	No details available	Wavelet, Gabor, Fourier	Atomic texture features	Random field decomposition	Local binary patterns	Edge statistics	Other	No details available	Edge direction histogram	Template matching	Fourier descriptors	Elastic models	Curvature scale space	Bounding box/ellipse	Elementary descriptors	Other	No details available	Layout	Face Detection	Keywords
ADL	[90]										*																		*
AltaVista	[2]									*							*									*			
Amore	[58]							*											*										*
ASSERT	[71]				*		*					*			*									*			*		
BDLP	[13]								$\star^2$																				*
Blobworld	[14]						*					*												*			*		
CANDID	[38]							*				*																	
C-bird	[46]		*					*	*3						*									*	$\star^4$		*		
CBVQ	[75]		*								*													*					i l
Chabot	[61]		*																										*
CHROMA	[43]							*																			*		
Circus	[66]		*									*																	
Compass	[11]		*	*											*														
Diogenes	[92]																											*	*
DrawSearch	[69]				*								*							*									
Excalibur	[22]									*							*									*			
FIDS	[8]		*								*			*	*												*		
FIR	[88]										*																		
FOCUS	[20]		*																					*			*		
FRIP	[39]				*							*												*	$\star^5$		*		
ImageFinder	[35]									*							*												
ImageMiner	[41]							*				*											*	*			*		
ImageRETRO	[86]		*		*				* <sup>6</sup>																$\star^7$				
ImageRover	[70]		*									*			*												*		*
ImageScape	[44]							*	* <sup>8</sup>			*		*						*				*			*	*	
iPURE	[1]				*								*							*			*	*			*		
Jacob	[15]		*									*			*														

KIWI	[48]				*						*					$*^{16}$											*		
LCPD	[45]													*		*9													
MARS	[62]		*								*	*								*							*		*
MetaSEEk	[7]		*									*																	*
MIDSS	[17]		*									*						*											
MIR	[79]		*	*																							*	*	*
NETRA	[51]						*				*									*							*		
Photobook	[64]	*											*								*							*	
Picasso	[9]						*														*		*	*			*		
PicHunter	[19]		*	*		*																							*
PicSOM	[42]				*											$\star^1$		*		*							*		
PicToSeek	[26]		*																						$\star^{10}$				
QBIC	[60]		*		*							*							*					*			*		*
Quicklook	[18]		*	*	*	*			*11		*	*			*			*						*			*		
RETIN	[24]							*			*																		
Shoebox	[55]				*				$\star^{12}$																				*
SIMBA	[72]															$\star^{13}$													
SMURF	[87]		*																						$*^{14}$				
SQUID	[56]																					*							
Surfimage	[59]	*	*								*	*				$\star^{15}$		*			*								
SYNAPSE	[53]															$\star^{16}$													
TODAI	[54]														*														*
Viper	[77]		*					*			*																		
VIR	[4]									*							*									*			*
VisualSEEk	[76]							*															*	*			*		
VP	[37]																								$\star^{17}$				
WebSEEk	[74]		*																										*
WebSeer	[80]		*						*18																			*	*
WISE	[89]										*																		

1 local probability

3 DCT of chromaticity

5 radius vector

7 number of regions, number of holes

9 projection histogram

11 spatial chromaticity histogram

13 histogram of polynomial of local intensities

2 number of regions with that color 4 generalized Hough transform

6 greyness, background proportion

8 Laplacian

10 angles between edges, and cross ratios of them

12 color channel variance

14 weighted point sets, polylines

15 intensity surface curvature histogram 17 convex parts

16 (multiscale) intensity surface derivatives 18 farthest neighbor histogram

Table 1: Overview of systems and their search features.

We have tried to group the low level features into meaningful categories. For example, various forms of choosing important colors are grouped into 'dominant colors'. The 'eigen image' is listed under color, because it is a feature derived from the global image color values, rather than local patterns as in texture. The 'atomic texture features' are features such as contrast, anisotropy, density, randomness, directionality, softness, and uniformity, often variations of the Tamura features [81], and often derived from a cooccurrence matrix of pixel values. The 'edge statistics' include the edge image histogram, edge orientation histogram, and others. Recall that we consider edge orientation over all pixels as texture, but edge orientation at only region contours as shape information, see section 2. The 'elementary shape descriptors' are features such as centroid, area, orientation and lenght of major and minor axes, eccentricity, circularity, and features derived from algebraic moments. The feature 'layout' is the absolute or relative spatial position of the color, texture, or shape features. Taking color histograms per fixed subimage is a simple way of exploiting spatial layout; considering the relative position (constellation) of regions is more complex.

Of the 56 systems in the table, 46 use any kind of color features, 38 use texture, 29 use shape, 20 layout, and 5 use face detection. Perhaps the reason for this order of frequencies of usage is their complexity and, coupled to that, effectiveness. The simpler the features can be extracted, the easier it is to implement it in the system, and the easier it is to use that feature class in the right way. For example, color features are often found effective, because good color features are not very difficult to design and implement. However, shape features that are for example robust against noise and occlusion, are more involved. As a result, only overly simple features such as area are used, which is often little effective.

## 5 Conclusions

Most systems are products of research, and therefore emphasize one aspect of content-based retrieval. Sometimes this is the sketching capability in the user interface, sometimes it is the new indexing data structure, etc. Some systems exist in a research version and in a commercial or production version. The commercial version is usually less advanced, and shows more standard searching capabilities. For example, a research version of Amore exhibits sketching and more fancy result visualization than is shown in the Arthur application system.

A number of systems provide a user interface that allows more powerful query formulation than is useful in the demo system. For example, if a user can paint a cartoon, but the database contains only airplanes, the system will always retrieve images of airplanes. For such a limited database, no powerful sketching interface is needed. Also, because most workstations have a mouse, but easy sketching needs a pencil, such a painting tool is often of little use. Drawing polygonal shapes with a mouse works well, however.

Most systems use color and texture features, few systems use shape feature, and still less use layout features. The retrieval on color usually yield images with similar colors. Retrieval on texture does not always yield images that have clearly the same texture, unless the database contains many images with a dominant texture. Searching on shape gives often suprising results. Apparently the shape features used for matching are not the most effective ones.

Indexing data structures are often not used. Indeed, for small collections of images, an indexing data structure is not needed, and a linear search can be sufficiently fast. Contemporary computers can perform simple matching of hundreds of images in near real time.

It is difficult to evaluate how successful content-based image retrieval systems are, in terms of effectiveness, efficiency, and flexibility. Of course there are the notions of precision (the ratio of relevant images to the total number of images retrieved) and recall (the percentage of relevant images among all possible relevant images). Many articles about systems give figures about precision and recall. Most of them are good, but hard to verify. One reason is that many hyperlinks on the Web are not active anymore, a design flaw of the Web. However, there are also considerations intrinsic to retrieval systems.

If the database only contains fighting airplanes, and the user can only ask for images similar to a chosen fighting airplanes, the system will successfully return fighting airplanes. If the domain is so narrow, it may make more sense to look for dissimilar images than for similar images. On the other hand, if the database is very diverse and contains only a single image of a chicken, asking for images similar to that chicken will not result in other chicken images. The larger the collection of images, the more chance that it contains an

image similar to the query image. The Web is a large enough test set, and free of charge, reason why some image retrieval systems exploit webcrawlers. Having a specific object in mind, looking for images with similar objects is a frustrating experience, however. Indeed, before you learn to walk, you crawl first.

It is widely recognized that most current content-based image retrieval systems work with low level features (color, texture, shape), and that next generation systems should operate at a higher semantic level. One way to achieve this is to let the system recognize objects and scenes. Although this is difficult in general, it should be feasible for specific domains. For example, the ImageMiner system classifies landscapes, and body plans have been used to recognize animals and people [23]. Once separate entities in images are recognized, the way to semantic reasoning lies open.

## References

- Gaurav Aggarwal, Pradeep Dubey, Sugata Ghosal, Ashutosh Kulshreshtha, and Abhinanda Sarkar. ipure: Perceptual and user-friendly retrieval of images. In *Proceedings of IEEE Conference on Multimedia and Exposition (ICME 2000)*, July 2000.
- [2] Altavista image search. http://www.altavista.com/sites/search/simage.
- [3] M. Amadasun and R. King. Textural features corresponding to textural properties. *IEEE Transactions on Systems, Man and Cybernetics*, 19(5):1264–1274, October 1989.
- [4] J. Bach, C. Fuller, A. Gupta, A. Hampapur, B. Gorowitz, R. Humphrey, R. Jain, and C. Shu. Virage image search engine: an open framework for image management. In *Proceedings of the SPIE, Storage* and Retrieval for Image and Video Databases IV, San Jose, CA, pages 76–87, February 1996.
- [5] D. H. Ballard. Generalized Hough transform to detect arbitrary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(2):111–122, 1981.
- [6] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. The R\*-tree: An efficient and robust access method for points and rectangles. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 322–331, 1990.
- [7] A. B. Benitez, M. Beigi, and S.-F. Chang. Using relevance feedback in content-based image metasearch. *IEEE Internet Computing*, 2(4):59–69, July/August 1998.
- [8] Andrew Berman and Linda Shapiro. A flexible image database system for content-based retrieval. *Computer Vision and Image Understanding*, 75(1/2):175–195, 1999.
- [9] A. Del Bimbo, M. Mugnaini, P. Pala, and F. Turco. Picasso: Visual querying by color perceptive regions. In *Proceedings of the 2nd International Conference on Visual Information Systems, San Diego, December '97*, pages 125–131, 1997.
- [10] Alberto Del Bimbo. Visual Information Retrieval. Morgan Kaufman, 1999.
- [11] R. Brunelli and O. Mich. Image retrieval by examples. *IEEE Transactions on Multimedia*, 2(3):164– 171, September 2000.
- [12] Jean Marie Buijs and Michael Lew. Visual learning of simple semantics in imagescape. In Huijsmans and Smeulders [34], pages 131–138.
- [13] Chad Carson and Virginia E. Ogle. Storage and retrieval of feature data for a very large online image collection. *IEEE Computer Society Bulletin of the Technical Committee on Data Engineering*, 19(4):19–27, December 1996.
- [14] Chad Carson, Megan Thomas, Serge Belongie, Joseph M. Hellerstein, and Jitendra Malik. Blobworld: A system for region-based image indexing and retrieval. In Huijsmans and Smeulders [34].

- [15] M. La Cascia and E. Ardizzone. Jacob: Just a content-based query system for video databases. In Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP96), May 7-10, '96, Atlanta, Georgia, 1996.
- [16] S.-K Chang, Q.Y. Shi, and C.Y. Yan. Iconic indexing by 2-d strings. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(3):413–428, May 1987.
- [17] J.Y. Chen, C. A. Bouman, and J. P Allebach. Multiscale branch and bound image database search. In Proceedings SPIE/IS&T Conference on Storage adn Retrieval for Image and Video Databases V, volume 3022, pages 133–144, 1997.
- [18] G. Ciocca, I. Gagliardi, and R. Schettini. Quicklook2: An integrated multimedia system. *International Journal of Visual Languages and Computing, Special Issue on Multimedia Databases and Image Communication*, in print.
- [19] Ingemar J. Cox, Matthew L. Miller, Thomas P. Minka, Thomas Papathomas, and Peter N. Yianilos. The Bayesian image retrieval system, PicHunter: Theory, implementation and psychophysical experiments. *IEEE Transactions on Image Processing (to appear)*, 2000.
- [20] M. Das, E. M. Riseman, and B. Draper. Focus: Searching for multi-colored objects in a diverse image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition '97*, pages 756–761, June 1997.
- [21] John P. Eakins and Margaret E. Graham. Content-based image retrieval, a report to the JISC technology application programme. Technical report, Institute for Image Data Research, University of Northumbria at Newcastle, UK, January 1999. http://www.unn.ac.uk/iidr/report.html.
- [22] Excalibur visual retrievalware. http://vrw.excalib.com/.
- [23] David A. Forsyth and Margaret M. Fleck. Body plans. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 678–683, 1997.
- [24] J. Fournier, M. Cord, and S. Philipp-Foliguet. Retin: A content-based image indexing and retrieval system. *Pattern Analysis and Applications*, 4(2/3):153–173, 2001.
- [25] E. Fredkin. Trie memory. Communications of the ACM, 3:490-499, 1960.
- [26] Theo Gevers and Arnold Smeulders. Pictoseek: Combining color and shape invariant features for image retrieval. *IEEE Transactions on Image Processing*, 9(1):102–119, January 2000.
- [27] L. Grewe and A. C. Kak. Interactive learning of a multi-attribute hash table classifier for fast object recognition. *Computer Vision and Image Understanding*, 61(3):387–416, 1995.
- [28] W. I. Grosky. Multimedia information systems. *IEEE Multimedia*, 1(1):12–24, 1994.
- [29] William I. Grosky, Ramesh Jain, and Rajiv Mehrotra, editors. *The Handbook of Multimedia Informa*tion Management. Prentice Hall, 1997.
- [30] V. N. Gudivada and V. V. Raghavan. Content-based image retrieval systems. *IEEE Computer*, 28(9):18–31, September 1995.
- [31] Kyoji Hirata, Yoshinori Hara, Naoki Shibata, and Fusako Hirabayashi. Media-based navigation for hypermedia systems. In Proceedings of the fifth ACM conference on Hypertext, November 14-18, '93, Seattle, WA, USA, pages 159–173, 1993.
- [32] Kyoji Hirata, Sougata Mukherjea, Yusaku Okamura, Wen-Syan Li, and Yoshinori Hara. Objectbased navigation: An intuitive navigation style for content-oriented integration environment. In *Proceedings of the eighth ACM conference on Hypertext, '97, Southampton, UK*, pages 75–86, 1997. http://journals.ecs.soton.ac.uk/~lac/ht97/.

- [33] M. Hu. Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, IT-8:179–182, 1962.
- [34] D. P. Huijsmans and A. W. M. Smeulders, editors. Visual Information and Information Systems, Proceedings of the Third International Conference VISUAL '99, Amsterdam, The Netherlands, June 1999, Lecture Notes in Computer Science 1614. Springer, 1999.
- [35] Imagefinder. http://attrasoft.com/abm3\_4.html.
- [36] R. Jain. Infoscopes: Multimedia information system. In B. Furht, editor, *Multimedia Systems and Techniques*, pages 217–253. Kluwer, 1996.
- [37] Fumikazu Kanehara, Shin'ichi Satoh, and Takashi Hamada. A flexible image retrieval using explicit visual instruction. In *Proceedings of the Third International Conference on Document Analysis Recognition, Montreal, Canada, August '95*, pages 175–178, 1995.
- [38] P. M. Kelly, T. M. Cannon, and D. R. Hush. Query by image example: the CANDID approach. In SPIE Vol. 2420, Storage and Retrieval for Image and Video Databases III, pages 238–248, 1995.
- [39] ByoungChul Ko, Jing Peng, and Hyeran Byun. Region-based image retrieval using probabilistic feature relevance learning. *Pattern Analysis and Applications*, 4(2/3):174–184, 2001.
- [40] H. F. Korth and A. Silberschatz. Database System Concepts. McGraw-Hill, 1991.
- [41] J. Kreyss, M. Röper, P. Alshuth, Th. Hermes, and O. Herzog. Video retrieval by still image analysis with ImageMiner. In Proceedings of IS&T/SPIE's Symposium on Electronic Imaging: Science & Technologie, 8-14 Feb. '97, San Jose, CA, 1997.
- [42] Jorma Laaksonen, Markus Koskela, Sami Laakso, and Erkki Oja. Picsom content-based image retrieval with self-organizing maps. *Pattern Recognition Letters*, 21:1199–1207, 2000.
- [43] T.-S. Lai. CHROMA: a Photographic Image Retrieval System. PhD thesis, School of Computing, Engineering and Technology, University of Sunderland, UK, January 2000. http://osiris.sund.ac.uk/~cs0sla/thesis/.
- [44] Michael Lew, Kim Lempinen, and Nies Huijmans. Webcrawling using sketches. In Proceedings of the 2nd International Conference on Visual Information Systems, San Diego, December '97, pages 77–84, 1997.
- [45] Michael S. Lew, D. P. Huijsmans, and Dee Denteneer. Content based image retrieval: KLT, projections, or templates. In Smeulders and Jain [73], pages 27–34.
- [46] Z. N. Li, O. R. Zaïane, and B. Yan. C-bird: Content-based image retrieval from image repositories using chromaticity and recognition kernel. Technical Report CMPT98-03, School of Computing Science, Simon Fraser University, Canada, February 1998. http://ftp.fas.sfu.ca/pub/cs/TR/1998/CMPT1998-03.ps.gz.
- [47] Z.N. Li, О. R. Zaïane, and Z. Tauber. Illumination invariance and object model in content-based image and video retrieval. Journal of Visual 1999. Communication and Image Representation, 10(3):219-244,September http://www.cs.sfu.ca/cs/undergrad/CM/CMPT365/CBS/CBIRD.pdf.
- [48] E. Loupias and s. Bres. Key point-based indexing for pre-attentive similarities: The kiwi system. *Pattern Analysis and Applications*, 4(2/3):200–214, 2001.
- [49] Etienne Loupias and Nicu Sebe. Wavelet-based salient points: Applications to image retrieval using color and texture features. In Advances in Visual Information Systems, Proceedings of the 4th International Conference, VISUAL 2000, Lecture Notes in Computer Science 1929, pages 223–232. Springer, 2000.

- [50] W. Y. Ma. NETRA: A Toolbox for Navigating Large Image Databases. PhD thesis, Dept. of Electrical and Computer Engineering, University of California at Santa Barbara, June 1997.
- [51] Wei-Ying Ma and B. S. Manjunath. Netra: A toolbox for navigating large image databases. *Multimedia Systems*, 7(3):184–198, 1999.
- [52] B. S. Manjunath. Image browsing in the alexandria digital library project. D-Lib Magazine, page http://www.dlib.org/dlib/august95/alexandria/08manjunath.html, August 1995.
- [53] R. Manmatha and S. Ravela. A syntactic characterization of appearance and its application to image retrieval. In *Proceedings of the SPIE conference on Human Vision and Electronic Imaging II, Vol,* 3016, San Jose, CA, Feb. '97, 1997.
- [54] S. Michel, B. Karoubi, J. Bigun, and S. Corsini. Orientation radiograms for indexing and identification in image databases. In *Proceedings Eusipco-96, European conference on signal processing*, pages 1693–1696, 1996.
- [55] Timothy J. Mills, David Pye, David Sinclair, and Kenneth R. Wood. Shoebox: A digital photo management system. Technical Report Technical Report 2000.10, AT&T Research, http://ftp.uk.research.att.com:/pub/docs/att/tr.2000.10.pdf, 2000.
- [56] F. Mokhtarian, S. Abbasi, and J. Kittler. Efficient and robust retrieval by shape content through curvature scale space. In Smeulders and Jain [73], pages 35–42.
- [57] Sougata Mukherjea, Kyoji Hirata, and Yoshinori Hara. Towards a multimedia world wide web information retrieval engine. In Sixth International WWW Conference, 7-11 April '97, Santa Clara, CA, USA, 1997. http://decweb.ethz.ch/WWW6/Technical/Paper003/Paper3.html.
- [58] Sougata Mukherjea, Kyoji Hirata, and Yoshinori Hara. Amore: A world wide web image retrieval engine. The WWW Journal, 2(3):115–132, 1999. http://www.baltzer.nl/www/contents/1999/2-3.html.
- [59] Chahab Nastar, Matthias Mitschke, Christophe Meilhac, and Nozha Boujemaa. Surfimage: A flexible content-based image retrieval system. In Proceedings of the ACM International Multimedia Conference, 12-16 September '98, Bristol, England, pages 339–344, 1998. http://www-rocq.inria.fr/\_nastar/MM98/.
- [60] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin. The qbic project: Quering images by content using color, texture, and shape. In Poceedings of the SPIE Conference on Storage and Retrieval for Image and Video Databases, 2-3 February '93, San Jose, CA, pages 173–187, 1993.
- [61] Virginia E. Ogle and Michael Stonebraker. Chabot: Retrieval from a relational database of images. *IEEE Computer*, 28(9):40–48, September 1995.
- [62] Michael Ortega, Yong Rui, Kaushik Chakrabarti, Sharad Mehrotra, and Thomas S. Huang. Supporting similarity queries in MARS. In *Proceedings of the 5th ACM International Multimedia Conference*, *Seattle, Washington*, 8-14 Nov. '97, pages 403–413, 1997.
- [63] P. Pala and S. Santini. Image retrieval by shape and texture. *Pattern Recognition*, 32(3):517–527, 1999.
- [64] Alex Pentland, Rosalind W. Picard, and Stanley Sclaroff. Photobook: Content-based manipulation of image databases. *International Journal of Computer Vision*, 18(3):233–254, June 1996.
- [65] Zoran Pečenović. Image retrieval using latent semantic indexing. Master's thesis, Department of Electrical Engineering, École Polytechnique Fédéral de Lausanne, 1997.

- [66] Zoran Pečenović. Finding rainbows on the internet. Technical report, Section of Communication Systems, École Polytechnique Fédéral de Lausanne, 1998.
- [67] Plato. Meno. Perseus Encyclopedia, Tuft University, http://www.perseus.tufts.edu/Texts/chunk\_TOC.grk.html, 380 B.C.
- [68] Yong Rui, Thomas S. Huang, and Shih-Fu Chang. Image retrieval: Current techniques, promising directions and open issues. *Journal of Visual Communication and Image Representation*, 10(1):1–23, March 1999.
- [69] E. Di Sciascio, G. Mingolla, and M. Mongiello. Content-based image retrieval over the web using query by sketch and relevance feedback. In Huijsmans and Smeulders [34], pages 123–130.
- [70] S. Sclaroff, L. Taycher, and M. La Cascia. Imagerover: A content-based image browser for the world wide web. In *Proceedings IEEE Workshop on Content-based Access of Image and Video Libraries*, *June* '97, 1997.
- [71] C.-R. Shyu, C. E. Brodley, A. C. Kak, A. Kosaka, A. Aisen, and L. Broderick. Assert: A physicianin-the-loop content-based retrieval system for HRCT image databases. *Computer Vision and Image Understanding*, 75(1/2):111–132, July/August 1999.
- [72] Sven Siggelkow and Hans Burkhardt. Fast invariant feature extraction for image retrieval. In Remco C. Veltkamp, Hans Burkhardt, and Hans-Peter Kriegel, editors, *State-of-the-Art in Content-Based Image and Video Retrieval*. Kluwer, 2001.
- [73] A. W. M. Smeulders and R. Jain, editors. Image Databases and Multi-Media Search, proceedings of the First International Workshop IDB-MMS'96, Amsterdam, The Netherlands. Amsterdam University Press, August 1996.
- [74] J. R. Smith. *Integrated Spatial and Feature Image Systems: Retrieval, Compression and Analysis.* PhD thesis, Graduate School of Arts and Sciences, Columbia University, February 1997.
- [75] J. R. Smith and S.-F. Chang. Automated image retrieval using color and texture. Technical Report CU/CTR 408-95-14, CTR, Columbia University, July 1995.
- [76] J. R. Smith and S.-F. Chang. Querying by color regions using the VisualSEEk content-based visual query system. In M. T. Maybury, editor, *Intelligent Multimedia Information Retrieval*. AAAI Press, 1997.
- [77] David McG. Squire, Wolfgang Müller, Henning Müller, and Thierry Pun. Content-based query of image databases: inspirations from text retrieval. *Pattern Recognition Letters*, 21:1193–1198, 2000.
- [78] Rohini Srihari. Automatic indexing and content-based retrieval of captioned images. *IEEE Computer*, 28(9):49–56, September 1995.
- [79] Rohini Srihari, Zhongfei Zhang, and Aibing Rao. Intelligent indexing and semantic retrieval of multimodal documents. *Information Retrieval*, 2(2):245–275, 2000. http://www.cse.buffalo.edu/~arao/Papers/ir.ps.gz.
- [80] Michael J. Swain, Charles Frankel, and Vassilis Athitsos. WebSeer: An image search engine for the world wide web. Technical Report TR-96-14, Deptartment of Computer Science, University of Chicago, July 1996.
- [81] H. Tamura, S. Mori, and T. Yamawaki. Texture features corresponding to visual perception. *IEEE Transactions on Systems, Man and Cybernetics*, 8(6):460–473, 1978.
- [82] Hideyuki Tamura and Naokazu Yokoya. Image database systems: A survey. *Pattern Recognition*, 17(1):29–43, 1984.

- [83] Leonid Taycher, Marco La Cascia, and Stan Sclaroff. Image digestion and relevance feedback in the ImageRover WWW search engine. In *Proceedings of the 2nd International Conference on Visual Information Systems, San Diego, December '97*, pages 85–94, 1997.
- [84] Remco C. Veltkamp and Michiel Hagedoorn. State-of-the-art in shape matching. In Michael Lew, editor, *Principles of Visual Information Retrieval*, pages 87–119. Springer, 2001. ISBN 1-85233-381-2.
- [85] J. Vendrig. Filter image browsing: a study to image retrieval in large pictorial databases. Master's thesis, Dept. Computer Science, University of Amsterdam, The Netherlands, http://carol.wins.uva.nl/~vendrig/thesis/, February 1997.
- [86] J. Vendrig, M. Worring, and A. W. M. Smeulders. Filter image browsing: Exploiting interaction in image retrieval. In Huijsmans and Smeulders [34], pages 147–154.
- [87] Jules Vleugels and Remco C. Veltkamp. Efficient image retrieval through vantage objects. Pattern Recognition, 35(1):69–80, 2002.
- [88] S. Volmer. Tracing images in large databases by comparison of wavelet fingerprints. In Proceedings of the 2nd International Conference on Visual Information Systems, San Diego, December '97, pages 163–172, 1997.
- [89] James Ze Wang, Gio Wiederhold, Oscar Firschein, and Sha Xin Wei. Wavelet-based image indexing techniques with partial sketch retrieval capability. In *Proceedings of the Fourth Forum on Research and Technology Advances in Digital Libraries, Washington D.C., May '97*, pages 13–24, 1997. http://www-db.stanford.edu/~wangz/project/imsearch/ADL97/.
- [90] Z. Wang, L. Hill, and T. Smith. Alexandria digital library metadata creator with extensible markup language. In Proceedings of the Third International Conference on Conceptions of Library and Information Science (CoLIS 3). Digital Libraries: Interdisciplinary Concepts, Challenges and Opportunities. Dubrovnik, Croatia, 1999.
- [91] D. White and R. Jain. Similarity indexing with the SS-tree. In *Proceedings of the 12th International Conference on Data Engineering, New Orleans, LA*, 1996.
- [92] C. T. Yu Y. A. Aslandogan. Multiple evidence combination in image retrieval: Diogenes searches for people on the web. In *Proceedings of 23rd Annual International ACM SIGIR Conference on Research* and Development in Information Retrieval (SIGIR 2000), pages 88–95, 2000.