C-BIRD: Content-Based Image Retrieval from Digital Libraries Using Illumination Invariance and Recognition Kernel

Ze-Nian Li, Osmar R. Zaïane, Bing Yan
School of Computing Science
Simon Fraser University
Burnaby, B.C., Canada V5A 1S6
{li, zaiane, bing}@cs.sfu.ca

Abstract

With huge amounts of multimedia information connected to the global information network (Internet), efficient and effective image retrieval from large image and video repositories has become an imminent research issue. This article presents our research in the C-BIRD (Content-Based Image Retrieval from Digital libraries) project. In addition to the use of common features such as keywords, color, texture, shape and their conjuncts, it is shown that (a) the color–channel–normalization enables Search by Illumination Invariance, and (b) the multi-level recognition kernel facilitates Search by Object Model in image and video databases.

1. Introduction

Image indexing and retrieval has lately drawn the attention of many researchers in the computer science community [6, 5, 3, 1, 7]. With the advent of the World-Wide Web (WWW), research in computer vision, artificial intelligence, databases, etc. is taken to a larger scale to address the problem of information retrieval from large repositories of images. Previous pattern recognition and image analysis algorithms in the vision and AI fields dealt with small sets of still images and did not scale. With large collections of images and video frames, scalability, speed and efficiency are the essence for the success of an image retrieval system.

There are two main families of image indexing and retrieval systems: those based on the content of the images (content-based) like color, texture, shape, objects, etc., and those based on the description of the images (description-based) like keywords, size, caption, etc. While description-based image retrieval systems are relatively easier to implement and to design user interface for, they suffer from the same problems as the information retrieval systems in text databases or Web search engines. It has been demonstrated that search engines using current methods based on simple keyword matching perform poorly. The precision of these search engines is very low and their recall is inadequate.

Content-based image retrieval systems [5, 3, 7] use visual features to index images. These systems differ mainly in the way they extract the visual features and index images, and the way they are queried. They give a relatively satisfactory result with regard to the visual clues, however, their precision and recall are still not optimized. Moreover, they lack the power of locating specific objects and identifying their details (size, position, orientation, etc.).

We have been developing the C-BIRD (Content-Based Image Retrieval from Digital libraries) system which combines automatically generated keywords and visual descriptors like color, texture, shape, and feature localization, to index images and videos in the World-Wide Web. This paper presents our results of *Search by Illumination Invariance*, and *Search by Object Model*.

Several color object recognition schemes exist that purport to take illumination change into account in an invariant fashion. In [2], we address the problem of illumination change by extending the original color histogram matching method by Swain and Ballard [8] to include illumination invariance in a natural and simpler way than heretofore. First, it is argued that a normalization on each color channel of the images is really all that is required to deal properly with illumination invariance. Second, with an aim of reducing the dimensionality of the feature space involved, a full–color (3D) representation is replaced by 2D chromaticity histograms. Third, the histograms are treated as images and undergone further compression in order to make them suitable for matching in large image and video databases.

For effective and efficient search by object model, the most important factors are the data representation (*modeling*) and the search (*matching*) strategy. In an effort to enable efficient retrieval in large image and video databases,

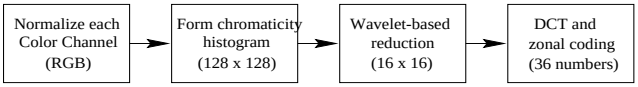


Figure 1. Achieving illumination invariance.

we have been developing a multi-level data-modeling and retrieval technique in C-BIRD. The technique uses a multi-level recognition kernel. Features of model objects are extracted at levels that are most appropriate to yield only the necessary yet sufficient details. It facilitates multi-level abstraction of the model and hence improves the matching efficiency and quality.

Compared to most existing approaches, our work has the following characteristics: (a) the exploration of color–channel–normalization for illuminant-invariant image and video retrieval, (b) the exploitation of intrinsic and compact feature vectors for search by object model, and (c) the multi-resolution recognition kernel for content-based image retrieval.

2. Illumination-Invariant Color Indexing

In this section it is shown as illustrated by Figure 1 that a simple color indexing method that is efficient and invariant under illuminant change can be derived for Search by Illumination Invariance.

A color–channel-normalization method was proposed in [2]. Given an image of size $m \times n$, each of the RGB channels is treated as a long vector of length $m \cdot n$. It is shown in [2] that by employing an L2 normalization on each of the three RGB vectors, the effect of any illumination change is approximately compensated. The color–channel–normalization step effectively accomplishes illumination invariance. The usage of chromaticity provides two additional advantages: (a) the color space is reduced from 3D (e.g., RGB, HSV, etc.) to 2D, hence less computations, (b) the chromaticity value is indeed guaranteed to be in the range of [0, 1.0]; this helps the formation of a small (well-bounded) 2D histogram space later.

From the chromaticity image, a chromaticity histogram can be obtained. This histogram itself is viewed as a *histogram-image* at the resolution of 128×128. Chromaticity histogram image matching without compression could be computationally intensive. A wavelet scaling function is applied several times to the original histogram–images to reduce its size down to 16×16. These reduced images undergo a 16×16 Discrete Cosine Transform (DCT). By experiment it is found [2] that using only the first 36 numbers in the upper left corner of the DCT coefficient matrix (a zonal coding method as defined in [9]) worked well.

Note that in this method only reduced, DCT transformed, quantized histogram–images are used — no inverse transforms are necessary and the indexing process is entirely carried out in the compressed domain.

3. Multi-level Recognition Kernel

This section describes a multi-resolution approach to modeling and matching for Search by Object Model.

3.1. Recognition kernel

A *recognition kernel* [4] is defined as a multi-resolution model for each object. Features of an object are extracted at levels that are most appropriate to yield only the necessary, yet sufficient details. Together they form the kernel.

Certain features (such as color) are known to be well-preserved under severe reduction of image resolution, they are hence used at low-resolution. Others (such as texture and shape) require relatively higher resolutions. As in [4], a three-level recognition kernel is employed (with Color 1 at the level of the lowest resolution, Color 2 and Texture 1 at the intermediate level, and Edge Orientations and Texture 2 at the level of the highest resolution).

1. Color 1: Colors in a model image are sorted according to their frequency (number of pixels) in the color histogram. The first few *Most Frequent Colors* (MFCs) and their frequencies are generally quite important as characteristic measures of an object. In this design, since color is used at a low resolution where only very few prominent colors are preserved, the MFCs become especially dominant.

2. Color 2: For each MFC, the centroid of all pixels is located first. Each pair of the centroids for two of the MFCs can be connected to produce an *MFC vector*. The length of the MFC vectors and the angles between them characterize the color distribution and shape of the object. To reduce the total number of MFC vectors, only the vectors that connect the first MFC centroid to the other MFC centroids are used. Hence, for k ($k \geq 2$) MFCs, the total number of MFC vectors is $k - 1$.

3. Texture 1: Edge density ("edgeness") is used to give an estimation whether the area is highly textured. Edge detection is only performed on the luminance image Y, where $Y = 0.299R + 0.587G + 0.114B$.

4. Edge Orientations: Similar to sorting colors, the edge orientations can also be sorted according to their frequency (number of pixels) and the *Most Frequent Orientations* (MFOs) can readily be obtained. The MFOs are especially useful in handling rotations. When an object is rotated on a 2-D plane (e.g., a book

is placed on the desk with a different orientation), all the edge orientations are simply incremented (or decremented) by a $\Delta\alpha$.

5. Texture 2: At this highest resolution for modeling, second order statistics could be used to generate texture feature vectors. In the current implementation, edge density and *MFO vectors* of the MFOs (derived similarly as in the MFC vectors) are used.

3.2. Matching

When the recognition kernel with a certain S is placed at a certain level in the image pyramid, features at the three levels will all be matched. A coarse-to-fine strategy is devised, namely, the search will start at the coarsest level of the kernel using colors only. After the MFCs are matched, the matching proceeds to the second level of the recognition kernel. Mainly, the MFC vectors will be checked to derive an improved estimation on the object size, location and orientation. Moreover, the edge density is also double-checked. The matching at the third level of the recognition kernel with the highest resolution is rather straight-forward. Since the size, location and orientation of a potential matching object is hypothesized at the previous matching steps, additional features of the recognition kernel such as the MFO vectors as defined in Section 3.1 at corresponding location and level will be examined. Since several more certainty factors C_i are introduced at each step to measure the degree of success in potential matching, a combined certainty factor $C = \prod_i C_i$ is defined. When C exceeds a selected threshold τ, the detection of an object is declared.

4. Implementation

Our image retrieval system C-BIRD has been implemented on both Unix and PC platforms. On both platforms, we used the same search engine and pre-processor written in C^{++}. One version of the user interface is implemented in Perl and HTML as a Web application, another version is implemented as a java applet. Figure 2 shows the general architecture for C-BIRD implementation. The system is accessible from http://jupiter.cs.sfu.ca/cbird/cbird.cgi, and http://jupiter.cs.sfu.ca/cbird/java/ (IE 4.0 or Netscape 4.0).

C-BIRD system rests on 4 major components:

- Extraction of images (Image Excavator);
- Processing of images to extract image features and storing precomputed data in a database (Pre-Processor);
- Querying (User Interface);

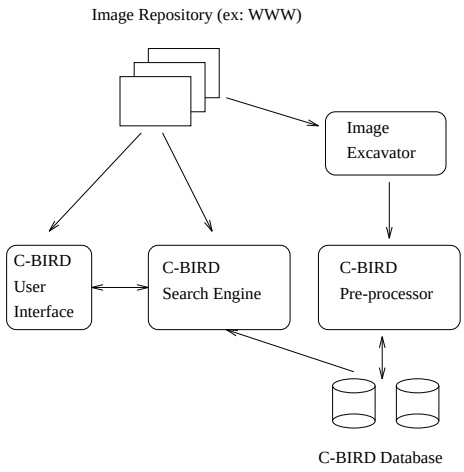


Figure 2. C-BIRD general architecture.

- Matching query with image features in the database (Search Engine).

The Image Excavator extracts images from an image repository. This repository can be the WWW space, in such case, the process crawls the Web searching for images, or a set of still images on disk or CD-ROM. Frames can also be extracted from video streams using cut-detection algorithms [10, 1] and processed as still images. Once images are extracted from the repository, they are given as input to the image analyzer (C-BIRD pre-processor) that extracts visual content features like color and edge characteristics. These visual features, along with the context feature like image URL, parent URL, keywords, etc., extracted with the Image Excavator, are stored in a database. The collection of images and the extraction of image features are processes that are done off-line before queries are submitted. When a query is submitted, accessing the original data in the image repository is not necessary. Only the precomputed data stored in the database is used for image feature matching. This makes C-BIRD more scalable and allows fast query responses for a large number of users and a huge set of images. When queries are submitted, only two processes are in action: the user interface interacting with users, and the search engine accessing and matching precomputed data. The user interface communicates with the search engine with a set of primitives. This allows to have different user interface implementations. The search engine accesses the database of the image visual and contextual features. If necessary, both the user interface and the search engine can access the images using their URL. We have implemented 8 types of searches and their combinations in C-BIRD:

1. Search by conjunctions / disjunctions of keywords;
2. Search by color histogram: similarity with color histogram in a sample image;

3. Search by illumination invariance: similarity with color chromaticity in a normalized sample image;

4. Search by color percentage: specification of up to 5 colors and percentages;

5. Search by color layout: specification of the layout of colors in a 1×1, 2×2, 4×4, or 8×8 grid;

6. Search by edge density and orientation;

7. Search by edge layout: specification of edge density and orientation in a 1×1, 2×2, 4×4, or 8×8 grid;

8. Search by object model: specification of an object to look for in images.

4.1. Retrieving the images from the WWW

The advantage of using the images available on the WWW is two-fold. First, the WWW provides us with a huge image repository which is a superb opportunity to test the efficiency and scalability of our implementation. Second, by using the images available on the WWW we can build an index for the WWW, that allows not only to find and retrieve images but also to find resources containing or referring to given images. Moreover, indexing images by sites can give interesting site content summaries by displaying thumbnail-sized images from a given Web site. Images from web pages are surprisingly representative of the associated textual content. Thus, browsing thumbnail-sized images from a site can give a broad idea about the content of the site.

To retrieve images from the WWW, we built a web spider (Excavator) that crawls the Web and downloads HTML pages and images. While images are analyzed by the pre-processor to extract content features, HTML pages are parsed to extract links to images and other HTML pages as well as descriptive information about images. When parsing a Web page, the Excavator extracts HTML IMG and EMBED tags and identifies image and video URLs. Subsequently, these images are downloaded and passed to the pre-processor.

Web pages on the WWW contain not only images, but also contextual information "describing" the images that can be extracted from text neighboring the images. The descriptive text can be used to educe keywords related to images. Being semi-structured, sections and components of an HTML pages can disclose valuable information about an image contained in the page.

Figure 3(a) illustrates the process of extracting images from the WWW.

Because the URL of the page containing an image is stored with the image meta-data, given an image, it is very easy to find the Web pages in which it is located.

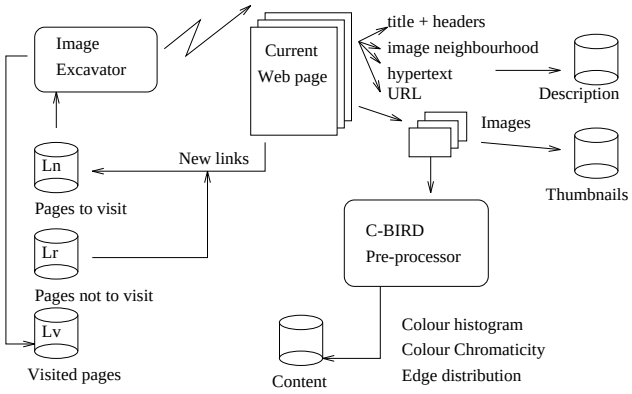


Figure 3. Excavator: The Web crawling process for image extraction.

4.2. C-BIRD database

The database used by C-BIRD is an addition to the image repository and contains mainly meta-data extracted by the pre-processor and the Image Excavator. As explained above, only features collected in this database at pre-processing time, are used by the search engine for image or image feature matching. During run time, minimal processing is done. For each image collected, the database contains some description information, a feature descriptor, and a layout descriptor, as well as a set of multi-resolution sub-images (i.e. search windows) feature descriptors. Neither the original image nor the sub-images are directly stored in the database; only their feature descriptors are stored.

The description information encompasses fields like: image file name, image URL, image type (i.e. gif, jpeg, bmp,...), list of all known web pages referring to the image (i.e. parent URLs), a list of keywords, and a thumbnail used by C-BIRD user interface for image and video browsing.

The feature descriptor is a set of vectors for each visual characteristic. The main vectors are: a chromaticity vector containing 36 values as described above in Section 2, a color vector containing the color histogram quantized to 256 colors ($8 \times 8 \times 4$ for $R \times G \times B$), MFC vector, and MFO vector. The MFC and MFO contain 5 color centroids and 5 edge orientation centroids for the 5 most frequent colors and 5 most frequent orientations (the edge orientations used are: $0°$, $22.5°$, $45°$, $67.5°$, $90°$, etc.). These centroids are used to derive the MFC and MFO vectors in the recognition kernel as presented in Section 3. The scalars "edge density" and "texture coarseness", used for texture estimation, are appended to the MFO vector.

The layout descriptor contains, a color layout vector, and an edge layout vector. These vectors allow to do matches with user defined layouts. Regardless of their original size,

all images are assigned an 8×8 grid. The most frequent colors for each of the 64 cells are stored in the color layout vector and the number of edges for each orientation in each of the cells is stored in the edge layout vector. The later is used for both search by edge density and search by edge orientation layout.

Since the recognition kernel searches for objects in each search window at a given resolution level, each sub-division (i.e. search window) is represented with a feature descriptor like the full image at the highest resolution level. These feature descriptors for the sub-images are stored with the image meta-data.

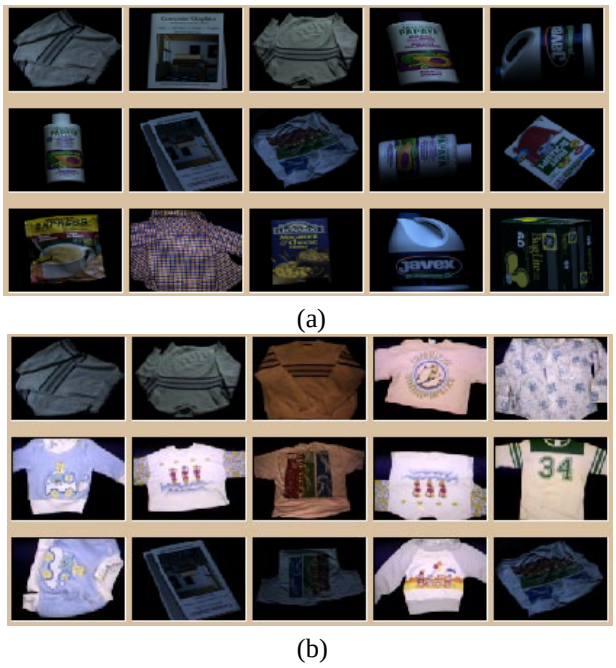
We use our illuminance invariant method to detect cuts in videos, and segment a video clip into frame sequences. The starting time and duration of the image sequence are stored with the meta-data. While the thumbnail is generated from the first frame, color and texture features are extracted from all frames.

4.3. Content-based retrieval results

C-BIRD on both platforms, has a simple and friendly user interface that allows querying by simple mouse clicks, browsing, and composing conjunctions of complicated queries. The current test database has over 1,300 images. The meta-data is stored in a SQL server running on a Pentium-II 333 MHz with 128 MB RAM. Search times are in the order of 0.1 to 2 seconds depending upon the type of search, except for the search by object, which may take up to 10 seconds to make comparisons in all sub-windows in the different resolutions and do all the necessary rotations. Notice that the search by object model begins by selecting only images that may potentially contain the object by shortlisting the images that contain the colors present in the object.

4.3.1. Search by illumination invariance

The experimental results for Search by Illumination Invariance are very promising. Figure 4 provides a comparison between the ordinary Search by Color Histogram and our Search by Illumination Invariance. The image sample selected for both searches is the first T-shirt image which is taken under a dim bluish light. The entire database is searched and the first 15 matches (sometimes mismatches) are shown in descending order of their matching scores. As expected, the by-color-histogram method (Figure 4(a)) is only capable of turning out many dark images, of which the third image happens to be a correct match (the same T-shirt being folded slightly differently and rotated). However, its matching score ranks behind a book. The result of by-illumination-invariance shown in Figure 4(b) is far better. All three occurrences of the sample T-shirt, the third one under a redish light, are found. Notably, it also finds many T-shirts under various illuminations. Since the sample T-shirt has basically two colors (dark stripes on white cloth), these matches are mostly correct in terms of their chromaticities, albeit unexpected.



(a)

(b)

Figure 4. Comparison of two results: (a) Result of Search by Color Histogram, (b) Result of Search by Illumination Invariance.

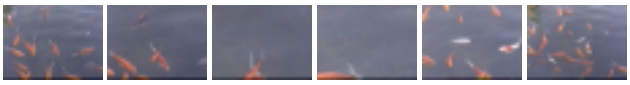


Figure 5. Selected frames of a video clip.

Figure 5 depicts some selected frames from a clip of goldfish scene in a 3-minute video that contains 22 cuts/clips. Because the camera was chasing the fish, the reflectance changes significantly. By selecting the threshold very carefully, the color histogram method still missed one cut and mistakenly added three more cuts (one of them at the third frame in the goldfish clip shown). As shown in [10] the color histogram is simply not able to satisfy both the precision and recall in video segmentation. Our illumination invariant method, however, detects all cuts correctly using a fixed threshold which works for other test videos as well.

4.3.2. Search by object model

Figure 6 illustrates an example of Search by Object Model. Figure 6(a) shows the eight book models of which the first book is selected. The object model in this case is a book. All 5 occurrences of this book with various sizes, positions and orientations are found (Figure 6(b)).

(a)

(b)

(c)

Figure 6. Result of Search by Object Model.

Figure 6(c) shows more detailed result of another search. The model book image and the centroids of the 5 MFCs are shown at the bottom of the figure. Among the 5 MFCs the color orange-red is the first MFC. As can be seen from the original image at the left, the sought book is at the upper-left quarter. The graphical display of the search window, the located book and locations of the color centroids are shown at the right of the figure. The book orientation ($44°$) and scale-ratio (1.09) are calculated using the weighted-average of the orientations and the lengths of the MFC vectors (vectors connecting the centroid of orange-red and the centroids of the other MFCs), respectively. Accordingly, the position (center of the book) is determined to be at (64, 79) which corresponds to the resolution of the bottom level of the recognition kernel. The search continues at the third level of the recognition kernel where the edge orientations and the MFO vectors are checked and confirmed.

5. Conclusion and Discussion

Content-based image retrieval is an important issue in the research and development of digital libraries which usually employ large multimedia databases. This paper presented our prototype system C-BIRD for content-based image retrieval from large image and video databases. Issues in both database design and image content based retrieval are addressed. Two new methods for image content based retrieval, i.e., Search by Illumination Invariance and Search by Object Model, are presented. At present, only models of 2D objects are supported. The further study on 3D modeling and matching is on the way.

Acknowledgments: This work was supported in part by the Canadian National Science and Engineering Research Council under the grant OGP-36727, and the grant for Telelearning Research Project in the Canadian Network of Centres of Excellence.

References

[1] P. Aigrain, H. Zhang, and D. Petkovic. Content-based representation and retrieval of visual media: A state-of-the-art review. *Int. J. Multimedia Tools and Applications*, 3:179–202, November 1996.

[2] M. Drew, J. Wei, and Z. Li. Illumination–invariant color object recognition via compressed chromaticity histograms of color–channel–normalized images. In *Proc. Int. Conf. on Computer Vision (ICCV '98)*, pages 533–540, 1998.

[3] C. Frankel, M. J. Swain, and V. Athitsos. Webseer: An image search engine for the world wide web. Technical Report 96-14, University of Chicago, Computer Science Department, August 1996.

[4] Z. Li and B. Yan. Recognition kernel for content-based search. In *Proc. IEEE Conf. on Systems, Man, and Cybernetics*, pages 472–477, 1996.

[5] M. Flickner, et al. Query by image and video content: the QBIC system. *IEEE Computer*, 28(9):23–32, 1995.

[6] A. Pentland, R. Picard, and S. Sclaroff. Photobook: Tools for content-based manipulation of image databases. In *SPIE Storage and Retrieval for Image and Video Databases II*, volume 2, 185, pages 34–47, San Jose, CA, 1994.

[7] J. Smith and S. Chang. Visually searching the web for content. *IEEE Multimedia*, 4(3):12–20, 1997.

[8] M. Swain and D. Ballard. Color indexing. *Int. J. Computer Vision*, 7(1):11–32, 1991.

[9] A. Tekalp. *Digital video processing*. Prentice Hall PTR, 1995.

[10] J. Wei, M. Drew, and Z. Li. Illumination invariant video segmentation by hierarchical robust thresholding. In *Proc. IS&T/SPIE Symp. on Electronic Imaging '98, Storage & Retrieval for Image and Video Databases VI, SPIE Vol. 3312*, pages 188–201, 1998.