EFFICIENT INDEXING, BROWSING AND RETRIEVAL OF IMAGE/VIDEO CONTENT

BY

YONG RUI

B.E., Southeast University, 1991
M.E., Tsinghua University, 1994

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1999

Urbana, Illinois

ABSTRACT

With advances in the computer technologies and the advent of the World Wide Web, there has been an explosion in the amount and complexity of digital data being generated, stored, transmitted, analyzed, and accessed. Much of this information is multimedia in nature, including digital images, video, audio, graphics, and text data. In order to make use of this vast amount of data, efficient and effective techniques to analyze and retrieve multimedia information based on its content need to be developed.

This thesis is dedicated to the two of the most important media types: images and videos. Novel approaches to feature analysis, content representation, indexing and retrieval are presented.

The main contributions of this thesis are: (i) reliable and robust feature extraction and representation techniques for images and videos; (ii) introduction of relevance feedback techniques to image retrieval, which greatly improves the retrieval performance and alleviates the user's query formulation burden; and (iii) introduction of a video hierarchical structure and construction of video Table of Contents to facilitate the video analysis and retrieval. Extensive experimental results over large test data sets have validated the proposed approaches.

To my parents and to Dongqin

ACKNOWLEDGMENTS

First of all, I would like thank my advisor Professor Thomas S. Huang for his advice and support through out my Ph.D. study. His open-mindedness has given me great freedom in exploring many challenging research topics and has made this thesis possible. I would like to thank my co-advisor, Professor Sharad Mehrotra, for his insightful guidance in the areas of database management and information retrieval. I thank my Ph.D. advisory committee members Professor Narendra Ahuja, Professor Kannan Ramchandran, and Professor Klara Nahrstedt for their inspiring and constructive discussions to set a high threshold for this thesis's. Special thanks to Professors Steve Levinson, Shih-Fu Chang, Brendan Frey, Pierre Moulin and Dan Roth for their valuable discussions in learning techniques.

I also would like to thank all my colleagues in the Image Formation and Processing Group, Beckman Institute for Advanced Science and Technology, and in the Database Management Systems Group, Department of Computer Science. In particular, I'd like to thank Greg Berry, Kaushik Chakrabarti, Jun Huang, Trausti Kristjansson, Qiong Liu, Milind Naphade, Mike Ortega, Kriengkrai Porkae, Alfred She, Roy R. Wang, Shaojun Wang, Kuan-Chieh Yen, Hu Yu, Sean X. Zhou, and Yueting Zhuang.

In countless ways I have received support and love from my family. Special thanks to my parents who themselves are electrical engineering professors and who led me into the field of EE since I was a child. Last but not least, deep thanks to my dear wife, Dongqin, for her understanding and love. Without her support, this thesis will never appear.

The research reported in this thesis was supported in part by NSF/DARPA/NASA DLI Program under Cooperative Agreement 94-11318, in part by ARL Cooperative Agreement No. DAAL01-96-2-0003, and in part by CSE Fellowship, UIUC.

One image collection used in the thesis is from the Museum Educational Site Licensing (MESL) project, sponsored by the Getty Information Institute. Another set of images were obtained from the Corel collection and used in accordance with their copyright statement.

TABLE OF CONTENTS

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

CHAPTER 1

INTRODUCTION TO VISUAL INFORMATION RETRIEVAL

In this chapter, we will first describe the background and state-of-the-art status of the area of visual information retrieval (VIR). We then introduce the research objectives and main contributions of the this thesis.

1.1 Background and Problem Statement

Recent years have seen a rapid increase of multimedia data. Every day, both military and civilian equipment generates gigabytes of digital data. Most of them are multimedia in nature, including images, graphics, videos, and animation, in addition to the traditional text data. Huge amounts of information are out there, and the expansion of the Internet is making this information accessible to everyone.

As a result, people are increasingly interested in using this information. However, before the information can be used, it must be located. Search engines exist for locating text information. Alta Vista, Lycos, and the like are among the most frequently visited Web sites, which indicates the need for such search engines. Unfortunately, search engines for data types other than text are yet to be developed.

The complexity of dealing with general multimedia data comes from various sources. Images and videos, for example, are much more information-rich compared with text documents. "An

image is worth a thousand words" is an underestimate. In addition, it is normally difficult to find the counterpart of keywords (as in the text domain) in image or video domains. Even more difficult is determining how to extract those "keyword counterparts" from images and videos, how to index them, and how to retrieve them in an efficient and effective manner.

This thesis tries to address some of these difficulties. In particular, this thesis addresses the problems associated with image and video media types, because we believe they are among the most widely used and most challenging media types in this research community.

1.2 State-of-the-Art Status

Since its advent, VIR has attracted many researchers from various research communities, including computer vision, image/video processing, library and information science, database management systems, etc.

A lot of research effort has been put into this area, ranging from government [1, 2], industry [3, 4, 5], to universities [6, 7, 8, 9, 10]. Even The International Standard Organization (ISO/IEC) has launched a new work item, MPEG-7 [11, 12, 13], to define a standard multimedia content description interface. Many special issues from leading journals have been dedicated to content-based image retrieval (CBIR) [14, 15, 16, 17] and many CBIR systems, both commercial [18, 3, 19, 4, 5] and academic [6, 7, 8, 9, 10], have been developed recently. For state-of-the-art surveys for image and video analysis and retrieval, please refer to [20, 21], respectively.

1.3 Research Objectives and Main Contributions

When this thesis was started in 1995, it was aimed to solve some of the challenging issues involved in VIR. Specifically,

- Image/video content analysis, which includes the feature extraction and feature representation

- Object (image/video) model, which can encapsulate the object features in a meaningful way
- Retrieval model for new media type (e.g. image and video)
- User interface that can take into account the characteristics of the new media types

Steady research achievements have been made along the above research directions over the years. Specifically, this thesis has made the following main contributions:

- Evaluation of various visual feature representations, including color, texture, and shape
- Development of a modified Fourier descriptor, which is both robust in representation and efficient in comparison
- Introduce of a multimedia object model that can encapsulate feature representations in a meaningful way
- Introduce the relevance feedback techniques into the retrieval process for new media types and development of an optimal soluation
- Introduce of a hierarchical video structure representation and development of an algorithm for video Table of Contents construction
- Development of novel techniques in video Index construction for video retrieval

Because most of the thesis research work is bound to the MARS project, we feel it is beneficial to first briefly describe the MARS project below.

1.3.1 The MARS project

The Multimedia Analysis and Retrieval System (MARS) project began in winter 1994 and involved researchers from Department of Electrical and Computer Engineering, Department of Computer Science, and School of Library and Information Science.

For the image collection, the data is provided by the Fowler Museum of Cultural History at the University of California-Los Angeles. It contains 286 ancient African and Peruvian artifacts and is part of the Museum Educational Site Licensing (MESL) Project, sponsored by the Getty Information Institute. After 1996, some more data sets are added to the test bed. Vistex data set is a collection of texture images, consisting a few thousand of images. It is obtained at MIT media lab at *ftp://whitechapel.media.mit.edu/pub/vistex/*. Corel dataset contains more than 70 000 images covering a wide range of more than 500 categories. The 120×80 resolution images are available at *http://corel.digitalriver.com/commerce/photostudio/catalog.htm*. This data set was obtained from the Corel collection and used in accordance with their copyright statement.

For the video collection, the video streams are MPEG compressed, with the digitization rate equal to 30 frames/s. To validate the effectiveness of the proposed approach, representatives of various movie types are used. Specifically, *The Bridges in Madison County (BMC)* (romantic-slow), *Pretty Woman (PW)* (romantic-fast), *Grease (GR)* (music), *The Mask (MS)* (comedy), *Star Trek (ST)* (science fiction-slow), *Star War (SW)* (science fiction-fast), and *Total Recall (TR)* (action) are used in our experiments. Each video clip lasts about 10-20 min.

We next give a brief description of MARS system architecture. For a detailed description, the reader is referred to [7, 22].

User interface. This is the interface where the user can interact with the system. The current implementation is written in Java applets and accessible over the Internet. The user interface allows users to graphically pose content-based queries as well as traditional text-based queries.

Retrieval subsystem. It takes the query specified at the user interface, evaluates the query using the information stored in all the three databases, and returns to the user images/videos that are best matches to the input query. The query language supported allows users to pose complex queries that are composed using low-level visual features as well as textual descriptions [22]. Another unique feature of the MARS retrieval subsystem is that it supports *relevance feedback* [23, 24]. Relevance feedback is the process

of automatically adjusting an existing query using the information fed back by the user about the relevance of previously retrieved objects such that the adjusted query is a better approximation to the information need of the user [25, 26]. This approach greatly reduces the user's effort of composing a query and captures the user's information need more precisely. Detailed description of relevance feedback technique is presented in Chapter 4.

Indexing subsystem. For large image collections, it is highly desirable to match queries without searching the entire image collection. To achieve this, multidimensional indexing techniques need to be utilized. The current MARS-image system uses a clustering-based indexing approach to facilitate the fast search. For detailed information, please refer to [27].

Feature extraction subsystem. MARS supports both global (whole image) features as well as local (image region) features. The features used in the system are color, texture, and shape [22]. Specifically, we use color histogram and color moments as the color feature representations [28]; coarseness-contrast-directionality, wavelet-based texture, and co-occurrence matrix texture as the texture representations [23]; and modified Fourier descriptors as the shape feature representation [29].

1.4 Dissertation Organization

To clearly present the content of the research, the thesis is divided into two parts. The first part, from Chapter 2 to Chapter 5, deals primarily with image media type. We first give a detailed review of the state-of-the-art of image retrieval in Chapter 2. In Chapter 3 we will describe the visual features and representations used in MARS for image retrieval. We then describe our proposed object model and retrieval model (enhanced with relevance feedback) in Chapter 4. Future research directions of image retrieval are given in Chapter 5.

Having the knowledge and techniques in Part I, we will address the video media type in Part II, from Chapter 6 to Chapter 11. We first give a review, in Chapter 6 on video analysis at

various levels, including shot boundary detection, key frame extraction, etc. We then present our proposed video analysis technique – video Table of Contents construction – in Chapter 7. Future research topics are suggested in Chapter 11.

PART I

PART I IMAGE SYSTEM

CHAPTER 2

INTRODUCTION TO IMAGE ANALYSIS AND RETRIEVAL

Among the various media types, images are of prime importance. Not only is it the most widely used media type besides text, but it is also one of the most widely used bases for representing and retrieving videos and other multimedia information.

Image retrieval has been a very active research area since the 1970s, with the thrust from two major research communities, database management and computer vision. These two research communities study image retrieval from different angles, one being text-based and the other visual-based.

Text-based image retrieval can be traced back to the late 1970s. A very popular framework of image retrieval at that time was to first annotate the images by text and then to use text-based database management systems (DBMS) to perform image retrieval. Representatives of this approach are [30, 31, 32, 33] and two comprehensive surveys on this topic are [34, 35]. Many advances, such as data modeling, multidimensional indexing, query evaluation, etc. have been made along this research direction. However, there are two major difficulties, especially when the size of image collections is large (tens or hundreds of thousands). One is the vast amount of labor required in manual image annotation. The other difficulty, which is more essential, results from the rich content in the images and the subjectivity of human perception. That is, different people may perceive the same image content differently. The perception subjectivity and annotation impreciseness may cause unrecoverable mismatches in the later retrieval processes.

In the early 1990s, because of the emergence of large-scale image collections, the two difficulties faced by the manual annotation approach became increasingly more acute. To overcome these difficulties, content-based image retrieval was proposed. That is, instead of being manually annotated by text-based keywords, images would be indexed by their own visual content, such as color, texture, etc. Since then, many techniques in this research direction have been developed and many image retrieval systems, both research and commercial, have been built. The advances in this research direction are mainly contributed by the computer vision community. Many special issues of leading journals have been dedicated to this topic [14, 15, 16, 36, 17].

This approach has established the general framework of modern image retrieval. However, many research issues remain to be solved before such retrieval systems can be put into practice. Regarding content-based image retrieval, we feel there is a need to survey what has been achieved in the past few years and what are the potential research directions that can lead to compelling applications.

There are three major bases for content-based image retrieval: visual feature extraction, multidimensional indexing, and retrieval system design. The remainder of this chapter is organized as follows. Section 2.1 reviews various visual features and their corresponding extraction and representation techniques. To facilitate fast search in large-scale image collections, effective indexing techniques need to be explored. Section 2.2 evaluates various such techniques, including dimension reduction and multidimensional indexing. State-of-the-art commercial and research systems and their distinct characteristics are described in Section 2.3.

2.1 Feature Extraction

Feature (content) extraction is the basis of content-based Image retrieval. In a broad sense, features may include both text-based features (keywords, annotations, etc.) and visual features (color, texture, shape, faces, etc.). However, because automatic generation of keywords from plain images is not possible at the current stage, we will confine ourselves in the techniques of visual feature extraction. Because of perception subjectivity, a single best presentation for

a given feature does not exist. As we will see, for any given feature, multiple representations exist that characterize the feature from different perspectives.

2.1.1 Color

Color feature is one of the most widely used visual features in image retrieval. It is relatively robust to background complication and independent of image size and orientation. Some representative studies of color perception and color spaces can be found in [37, 38, 39].

In image retrieval, color histogram is the most commonly used color feature representation. Statistically, it denotes the joint probability of the intensities of the three color channels. Swain and Ballard proposed histogram intersection, an L_1 metric, as the similarity measure for the color histogram [28]. To take into account the similarities between similar but not identical colors, Ioka [40] and Niblack et al. [18] introduced an L_2-related metric for comparing the histograms. Furthermore, considering that most color histograms are very sparse and thus sensitive to noise, Stricker and Orengo [41] proposed to use the cumulated color histogram. Their research results demonstrated the advantages of the proposed approach over the conventional color histogram approach [41].

Besides color histogram, several other color feature representations have been applied in image retrieval, including color moments and color sets. To overcome the quantization effects as in color histogram, Stricker and Orengo proposed using the color moments approach [41]. The mathematical foundation of this approach is that any color distribution can be characterized by the its moments. Furthermore, because most of the information is concentrated on the low-order moments, only the first moment (mean), and the second and third central moments (variance and skewness) were extracted as the color feature representation. Weighted Euclidean distance was used to calculate the color similarity.

To facilitate a fast search over large-scale image collections, Smith and Chang proposed color sets as an approximation to color histogram [42, 43]. They first transformed the (R,G,B) color space into a perceptually uniform space, such as hue-saturation-value (HSV), and then

quantized the transformed color space into M bins. A color set is defined as a selection of the colors from the quantized color space. Because color set feature vectors were binary, a binary search tree was constructed to allow a fast search. The relationship between the proposed color sets and the conventional color histogram was further discussed [42, 43].

2.1.2 Texture

Texture refers to the visual patterns with properties of homogeneity that do not result from the presence of a single color or intensity [44]. Texture is an innate property of virtually all surfaces, including clouds, trees, bricks, hair, fabric, etc. It contains important information about the structural arrangement of surfaces and their relationship to the surrounding environment [45]. Because of the importance and usefulness of texture in pattern recognition and computer vision, research results in the past three decades have been rich. Now, texture finds its way in image retrieval. More and more research achievements are being added to the texture representations.

In the early 1970s, Haralick et al. proposed the co-occurrence matrix representation of texture feature [45]. This approach explored the gray level spatial dependence of texture. It first constructed a co-occurrence matrix based on the orientation and distance between image pixels and then extracted meaningful statistics from the matrix as the texture representation. Many other researchers followed the same line and further proposed enhanced versions. For example, Gotlieb and Kreyszig studied the statistics originally proposed in [45] and experimentally found out that *contrast*, *inverse deference moment*, and *entropy* had the biggest discriminatory power [46].

Motivated by the psychological studies in human visual perception of texture, Tamura et al. explored the texture representation from a different angle [47]. They developed computational approximations to the visual texture properties found to be important in psychology studies. The six visual texture properties were *coarseness, contrast, directionality, linelikeness, regularity*, and *roughness*. One major distinction between the Tamura texture representation and the

co-occurrence matrix representation is that all the texture properties in Tamura representation are visually meaningful whereas some of the texture properties used in co-occurrence matrix representation may not be. This characteristic makes the Tamura texture representation very attractive in image retrieval, as it can provide a more friendly user interface. The Query By Image Content (QBIC) system [48] and the MARS system [7, 49] further improved this texture representation.

In the early 1990s, after wavelet transform was introduced and its theoretical framework established, many researchers began to study the use of wavelet transform in texture representation [50, 51, 52, 53, 54, 55]. Smith and Chang [50, 44] used the statistics (mean and variance) extracted from the wavelet subbands as the texture representation. This approach achieved over 90% accuracy on the 112 Brodatz texture images. To explore the middle-band characteristics, a tree-structured wavelet transform was used by Chang and Kuo [51] to further improve the classification accuracy. Wavelet transform was also combined with other techniques to achieve better performance. Gross et al. used wavelet transform, together with Karhunen-Loeve (KL) expansion and Kohonen maps, to perform texture analysis in [53]. Thyagarajan et al. [55] and Kundu and Chen [54] combined wavelet transform with co-occurrence matrix to take advantage of both statistics-based and transform-based texture analysis.

There are quite a few review papers in this area. An early review paper by Weszka et al. compared the texture classification performance of Fourier power spectrum, second-order gray level statistics (co-occurrence matrix), and first-order statistics of gray level differences [56]. They tested the three methods on two sets of terrain samples and concluded that Fourier method performed poorly, and the other two were comparable. In [57], Ohanian and Dubes compared and evaluated four types of texture representations, namely Markov random field representation [58], multichannel filtering representation, fractal based representation [59], and co-occurrence representation. They tested the four texture representations on four test sets, two synthetic (fractal and Gaussian Markov random field) and two natural (leather and painted surfaces). They discovered that co-occurrence matrix representation performed best in their test

sets. In a more recent paper, Ma and Manjunath [60] evaluated the texture image annotation by various wavelet transform representations, including orthogonal and bi-orthogonal wavelet transforms, tree-structured wavelet transform, and Gabor wavelet transform. They found that the Gabor transform was the best among the tested candidates, matching the human vision study results [44].

2.1.3 Shape

An important criterion for shape feature representation is that it should be invariant to translation, rotation, and scaling, since human beings tend to ignore such variations for recognition and retrieval purpose. In general, the shape representations can be divided into two categories, boundary-based and region-based. The former uses only the outer boundary of the shape, and the latter uses the entire shape region [61]. The most successful representatives for these two categories are Fourier descriptor and moment invariants.

The main idea of Fourier descriptor is to use the Fourier transformed boundary as the shape feature. Some early work can be found in [62, 63]. To take into account the digitization noise in the image domain, Rui et al. proposed a modified Fourier descriptor that is both robust to noise and invariant to geometric transformations [61].

The main idea of moment invariants is to use region-based moments, which are invariant to transformations, as the shape feature. Hu [64] identified seven such moments. Based on his work, many improved versions emerged. Based on the discrete version of Green's theorem, Yang and Albregtsen [65] proposed a fast method of computing moments in binary images. Motivated by the fact that most useful invariants were found by extensive experience and trial and error, Kapur et al. developed algorithms to systematically generate and search for a given geometry's invariants [66]. Realizing that most researchers did not consider what happened to the invariants after image digitization, Gross and Latecki developed an approach that preserved the qualitative differential geometry of the object boundary, even after an image was digitized [66]. In [67, 68], a framework of algebraic curves and invariants is proposed to

represent complex objects in cluttered scene by parts or patches. Polynomial fitting is done to represent local geometric information, from which geometric invariants are used in object matching and recognition.

There are some recent work in boundary-based shape representation and matching, including Chamfer matching [69, 70], Turning function [71], and wavelet descriptor [72]. Barrow et al. first proposed the Chamfer matching technique, which compared two collections of shape fragments at a cost proportional to linear dimension, rather than area [69]. In [70], to speed up the Chamfer matching process, Borgerfos proposed a hierarchical Chamfer matching algorithm. The matching was done at different resolutions, from coarse to fine. Along a similar line of Fourier descriptor, Arkin et al. developed a Turning-function-based method for comparing both convex and concave polygons [71]. In [72], Chuang and Kuo used wavelet transform to describe object shape. It embraced the desirable properties such as multiresolution representation, invariance, uniqueness, stability, and spatial localization.

Some recent review papers in shape representations are [73, 74]. In [73], Li and Ma showed that the geometric moments method (region-based) and the Fourier descriptor (boundary-based) were related by a simple linear transformation. In [74], Mehtre et al. compared the performance of boundary-based representations (chain code, Fourier descriptor, Universidade Nova de Lisboa (UNL) Fourier descriptor), region-based representations (moment invariants, Zernike moments, pseudo-Zernike moments), and combined representations (moment invariants and Fourier descriptor, moment invariants and UNL Fourier descriptor). Their experiments showed that the combined representations outperformed the simple representations.

In addition to two-dimensional (2D) shape representations, there were many methods developed for three-dimensional (3D) shape representations. In [75], Wallace and Wintz presented a technique for normalizing Fourier descriptors that retained all shape information and that was computationally efficient. They also took advantage of an interpolation property of Fourier descriptor that resulted in efficient representation of 3D shapes. In [76], Wallace and Mitchell proposed using a hybrid structural/statistical local shape analysis algorithm for 3D shape rep-

resentation. Further, Taubin proposed to use a set of algebraic moment invariants to represent both 2D and 3D shapes [77], which greatly reduced the computation required for shape matching.

2.1.4 Color layout

Although the global color feature is easy to calculate and can provide reasonable discriminating power in image retrieval, it tends to give too many false positives when image collection is large. Many research results suggested that using color layout (both color feature and spatial relations) is a better solution to image retrieval. To extend the global color feature to a local one, a natural approach is to divide the whole image into subblocks and extract color features from each of the subblocks [19, 78]. A variation of this approach is the quad-tree based color layout approach [79], where the entire image was split into quad-tree structure and each tree branch had it own histogram to describe its color content. Although conceptually simple, this regular-subblock-based approach cannot provide accurate local color information and is computation - and storage - expensive. A more sophisticated approach is to segment the image into regions with salient color features by color set back-projection and then to store the position and color set feature of each region to support later queries [42]. The advantage of this approach is its accuracy while the disadvantage is the general difficulty of performing reliable image segmentation.

To achieve a good trade-off between the above two approaches, several other color layout representations were proposed. In [80], Rickman and Stonham proposed a color tuple histogram approach. They first constructed a code book that described every possible combination of coarsely quantized color hues that might be encountered within local regions in an image. Then a histogram-based on quantized hues was constructed as the local color feature. In [81], Stricker and Dimai extracted the first three color moments from five predefined partially overlapping fuzzy regions. The use of the overlapping region made their approach relatively insensitive to small regions transformations. In [82], Pass et al. classified each pixel of a particular color as

either coherent or incoherent, based on whether or not it is part of large similarly colored region. By using this approach, widely scattered pixels were distinguished from clustered pixels, thus improving the representation of local color features. In [83], Huang et al. proposed what they called a color "correlogram" based color layout representation. They first constructed a color co-occurrence matrix and then used the auto-correlogram and correlogram as the similarity measures. Their experimental results showed that this approach was more robust than the conventional color histogram approach in terms of retrieval accuracy [83].

Along the same line of color layout feature, the layout of texture and other visual features can also be constructed to facilitate more advanced image retrieval.

2.1.5 Segmentation

Segmentation is very important to image retrieval. Both the shape feature and the layout feature depend on good segmentation. In this subsection we will describe some existing segmentation techniques used in both computer vision and image retrieval.

In [84], Lybanon et al. researched the morphological operation (opening and closing) based approach in image segmentation. They tested their approach in various types of images, including optical astronomical images, infrared ocean images, and magnetograms. While this approach was effective in dealing with the above scientific image types, its performance needs to be further evaluated for more complex natural scene images. In [85], Hansen and Higgins exploited the individual strengths of watershed analysis and relaxation labeling. Since a fast algorithm exists for watershed, they first used watershed to subdivide an image into catchmen basins. They then used relaxation labeling to refine and update the classification of catchmen basins initially obtained from watershed to take advantages of relaxation labeling's robustness to noise. In [86], Li et al. proposed a fuzzy-entropy-based segmentation approach. This approach is based on the fact that local entropy maxima correspond to the uncertainties among various regions in the image. This approach was very effective for images whose histogram do not

have clear peaks and valleys. Other segmentation techniques based on Delaunay triangulation, fractals, and edge flow can be found in [87, 88, 89].

All the above-mentioned algorithms are automatic. A major advantage of this type of segmentation algorithm is that it can extract boundaries from large numbers of images without occupying the user's time and effort. However, in an unconstrained domain, for non-preconditioned images, the automatic segmentation is not always reliable. What an algorithm can segment in this case is only regions, not objects. To obtain high-level objects, which is desirable in image retrieval, human assistance is needed.

In [90], Samadani and Han proposed a computer-assisted boundary extraction approach, which combined manual inputs from the user with the image edges generated by the computer. In [91], Daneels et al. developed an improved method of active contours. Based on the user's input, the algorithm first used a greedy procedure to provide fast initial convergence. Secondly, the outline was refined by using dynamic programming. In [92], Rui et al. proposed a segmentation algorithm based on clustering and grouping in spatial-color-texture space. The user defines where the attractor (object of interest) is, and the algorithm groups regions into meaningful objects.

Last comment worth mentioning in segmentation is that the requirements of segmentation accuracy are quite different for shape features and layout features. For the former, accurate segmentation is highly desirable, whereas for the latter, a coarse segmentation suffices.

2.1.6 Summary

As we can see from the above descriptions, many visual features have been explored, both previously in computer vision applications and currently in image retrieval applications. For each visual feature, there exist multiple representations, which model the human perception of that feature from different perspectives.

What features and representations should be used in image retrieval depends on the application. There is a need to develop an image-content description (model) to organize the features.

The features not only should be associated with the images but also should be invoked at the right place and time, whenever they are needed to assist retrieval. The details of this aspect is discussed in Chapter 4.

2.2 High-Dimensional Indexing

To make the content-based image retrieval truly scalable to large size image collections, efficient multidimensional indexing techniques need to be explored. There are two main challenges in such an exploration for image retrieval:

- High dimensionality: The dimensionality of the feature vectors is normally of the order of 10^2.
- Non-Euclidean similarity measure: Because Euclidean measures may or may not effectively simulate human perception of a certain visual content, other similarity measures, such as histogram intersection, cosine, and correlation need to be supported.

Toward solving these problems, one promising approach is to first perform dimension reduction and then use appropriate multidimensional indexing techniques, which can support non-Euclidean similarity measures.

2.2.1 Dimension reduction

Even though the dimension of the feature vectors in image retrieval is normally very high, the *embedded dimension* is much lower [93, 94]. Before utilizing any indexing technique, it is beneficial to first perform dimension reduction. There are at least two approaches in the literature: Karhunen-Loeve Transform (KLT) and columnwise clustering.

KLT, its variation in face recognition (eigenimage), and its variation in information analysis (principal component analysis (PCA)) have been studied by researchers in performing dimension reduction. In [95], Ng and Sedighian followed the eigenimage approach to carry out the dimension reduction, and in [96] Faloutsos and Lin proposed a fast approximation to KLT to

perform the dimension reduction. Experimental results showed that most real data sets (visual feature vectors) can be considerably reduced in dimension without significantly degrading retrieval quality [95, 96, 94]. Recently, Chandrasekaran et al. developed a low-rank singular value decomposition (SVD) update algorithm which was efficient and numerically stable in performing KLT [97]. Considering that the image retrieval system is a dynamic system and new images are continuously added to the image collection, a dynamic update of indexing structure is indispensably needed. This algorithm provides such a tool.

In addition to KLT, clustering is another powerful tool in performing dimension reduction. Clustering technique is used in various disciplines such as pattern recognition [98], speech analysis [99], and information retrieval [26], etc. Normally, it is used to cluster similar objects (patterns, signals, and documents) together to perform recognition or grouping. This type of clustering is called row-wise clustering. However, clustering can also be used in columnwise clustering to reduce the dimensionality of the feature space [26]. Experiments show that this is a simple and effective approach.

2.2.2 Multidimensional indexing techniques

After we identify the *embedded dimension* of the feature vectors, we need to select appropriate multidimensional indexing algorithms to index the reduced but still high dimensional feature vectors. There are three major research communities contributing in this area, i.e., computational geometry, database management, and pattern recognition. The existing popular multidimensional indexing techniques include bucketing algorithm, k-d tree, priority k-d tree [94], quad-tree, K-D-B tree, hB-tree, R-tree, and its variants R^+-tree and R^*-tree [100, 101, 102, 103, 27]. In addition to the above approaches, clustering and Neural Nets, widely used in pattern recognition, are also promising indexing techniques [98, 104].

The history of multidimensional indexing techniques can be traced back to middle 1970s, when cell methods, quad-tree, and k-d tree were first introduced. However, their performances were far from satisfactory. Pushed by the then-urgent demand of spatial indexing from ge-

ographical information systems (GIS) and computer aided design (CAD) systems, Guttman proposed the R-tree indexing structure in 1984 [100]. Based on his work, many other variants of R-tree were developed. Sellis et al. proposed R^+ tree in [101]. Greene proposed her variant of R-tree in [102]. In 1990, Beckmann et al. proposed the best dynamic R-tree variant, R^*-tree [103]. However, even for R^*-tree, it was not scalable to dimensions higher than 20 [19].

Very good reviews and comparisons of various indexing techniques in image retrieval can be found in [94, 95]. The research goal of White and Jain [94] was to provide general-purpose and domain-independent indexing algorithms. Motivated by k-d tree and R-tree, they proposed VAM k-d tree (they called it VAM because (1) its split orientation is based on the *variance*, and (2) the split position is *approximately* the *mean*.) and VAMSplit R-tree. Experimentally, they found that the VAMSplit R-tree provided the best performance, but the trade-off is the loss of dynamic nature of R-tree. In [95], Ng and Sedighian proposed a three-step strategy toward image retrieval indexing: reduce dimensions, evaluate existing indexing approaches, and customize the selected indexing approach. After dimension reduction with the eigenimage approach, the following three characteristics of the dimension-reduced data can be used to select good existing indexing algorithms:

- The new dimension components are ranked by decreasing variance
- The dynamic ranges of the dimensions are known
- The dimensionality is still fairly high

On their test data sets, they found that BA-KD-tree gave the best performance.

Considering that most of the tree indexing techniques were designed for traditional database queries (point queries and range queries) but not for the similarity queries used in image retrieval, there was a need to explore the new characteristics and requirements for indexing structures in image retrieval. Such a technique was explored in [105], where Tagare developed a tree adaptation approach that refined the tree structure by eliminating inefficient tree nodes for similarity queries.

So far, the above approaches only concentrated on how to identify and improve indexing techniques that are scalable to high dimensional feature vectors in image retrieval. The other nature of feature vectors in image retrieval, i.e., non-Euclidean similarity measures, has not been deeply explored. The similarity measures used in image retrieval may be non-Euclidean and may even be nonmetric. There are two promising techniques toward solving this problem: clustering and Neural Nets. Charikar et al. [106] proposed an incremental clustering technique for dynamic information retrieval. This technique had three advantages: a dynamic structure, the capability to handle high-dimensional data, and the potential to deal with non-Euclidean similarity measures. Rui et al. [27] further extended this technique in the direction of supporting non-Euclidean similarity measure and faster and more accurate search strategies.

Zhang and Zhong [104] proposed using self-organization map (SOM) Neural Nets as the tool for constructing the tree-indexing structure in image retrieval. The advantages of using SOM were its unsupervised learning ability, dynamic clustering nature, and the potential of supporting arbitrary similarity measures. Their experimental results over the Brodatz texture collection demonstrated that SOM was a promising indexing technique.

A more recent approach (hybrid tree) was proposed by Chakrabarti and Mehrotra [107]. A hybrid tree combines positive aspects of bounding region-based data structures (e.g., R-tree) and space partitioning data structures (e.g., KDB-tree) into a single data structure. It is more scalable to high dimensionalities and supports queries based on arbitrary distance functions.

2.3 Image Retrieval Systems

Since the early 1990s, content-based image retrieval has been a very active research area. Many image retrieval systems, both commercial and research, have been built. Most image retrieval systems support one or more of the following options [108]:

- Random browsing
- Search by example
- Search by sketch

- Search by text (including keyword or speech)
- Navigation with customized image categories

We have a rich set of search options today, but systematic studies involving actual users in practical applications still need to be done to explore the trade-offs among the different options listed above. Here, we will select a few representative systems and highlight their distinct characteristics.

2.3.1 QBIC

QBIC [18, 3, 19, 48, 109, 110, 91] is the first commercial content-based image retrieval system. Its system framework and techniques have profound effects on later image retrieval systems.

QBIC supports queries based on example images, user-constructed sketches and drawings, selected color and texture patterns, and the like. The color features used in QBIC are the average (R,G,B), (Y,i,q), (L,a,b), and MTM (mathematical transform to Munsell) coordinates, and a k element color histogram [19]. Its texture feature is an improved version of the Tamura texture representation [47], i.e., combinations of coarseness, contrast, and directionality [48]. The shape feature of QBIC consists of shape area, circularity, eccentricity, major axis orientation, and a set of algebraic moments invariants [109, 19]. QBIC is one of the few systems that utilized a high-dimensional indexing technique to facilitate fast searches. In its indexing subsystem, KLT is first used to perform dimension reduction and then R^*-tree is used as the multidimensional indexing structure [110, 19]. In its new system, text-based keyword search can be combined with content-based similarity search. The on-line QBIC demo is at *http://wwwqbic.almaden.ibm.com/*.

2.3.2 Virage

Virage is a content-based image search engine developed at Virage Inc. Similar to QBIC, Virage [4, 111] supports visual queries based on color, composition (color layout), texture, and

structure (object boundary information). But Virage goes one step further than QBIC. It also supports arbitrary combinations of the above four atomic queries. The users can adjust the weights associated with the atomic features according to their own emphasis. In [4], Bach et al. further proposed an open framework for image management. They classified the visual features ("primitive") as general (such as color, shape, or texture) and domain specific (face recognition, cancer cell detection, etc.). Various usefully "primitives" can be added to the open structure depending on the domain requirements. To go beyond the query-by-example mode, Gupta and Jain proposed a nine-component *query language* framework in [111]. The corresponding demos of Virage are at *http://www.virage.com/cgi-bin/query-e*.

2.3.3 RetrievalWare

RetrievalWare is a content-based image retrieval engine developed by Excalibur Technologies Corp. [5, 112]. From one of its early publications, we can see that its emphasis was in Neural Nets to image retrieval [5]. Its more recent search engine uses color, shape, texture, brightness, color layout, and aspect ratio of the image, as the query features [112]. It also supports the combinations of these features and allows the users to adjust the weights associated with each feature. Its demo page is available at *http://vrw.excalib.com/cgi-bin/sdk/cst/cst2.bat*.

2.3.4 Photobook

Photobook [6] is a set of interactive tools for browsing and searching images developed at MIT Media Lab. Photobook consists of three sub-books, from which shape, texture, and face features are extracted respectively. Users can then query based on corresponding features in each of the three sub-books.

In its more recent version of Photobook, FourEyes, Picard and colleagues proposed to include human in the image annotation and retrieval loop [113, 114, 115, 116, 117, 118, 119, 120]. The motivation of this was based on the observation that there was no single feature which can best model images from each and every domain. Furthermore, a human's perception is subjective.

They proposed a "society of model" approach to incorporate the human factor. Experimental results show that this approach is effective in interactive image annotation [114, 119].

2.3.5 VisualSEEk and WebSEEk

VisualSEEk [121, 122] is a visual feature search engine and WebSEEk [8] is a World-Wide-Web-oriented text/image search engine. Both were developed at Columbia University. The main research features are spatial relationship query of image regions and visual feature extraction from compressed domain [123, 124, 125, 126].

The visual features used in these systems are color set and wavelet transform based texture features [42, 43, 50, 44]. To speed up the retrieval process, these systems also contain binary tree-based indexing algorithms [127, 128, 129, 130].

VisualSEEk supports queries based on both visual features and their spatial relationships. This enables a user to submit a "sunset" query as red-orange color region on top and blue or green region at the bottom as its "sketch." WebSEEk consists of three main modules: image/video collecting module, subject classification and indexing module, and search, browse, and retrieval module. It supports queries based on both keywords and visual content. On-line demos are available at *http://www.ee. columbia.edu/~sfchang/demos.html.*

2.3.6 Netra

Netra is a prototype image-retrieval system developed in the University of California at Santa Barbara (UCSB) Alexandria Digital Library (ADL) project [9]. Netra uses color, texture, shape, and spatial location information in the segmented image regions to search and retrieve similar regions from the database. Main research features of the Netra system are its Gabor-filter-based texture analysis [131, 60, 132, 133], Neural-Nets-based image thesaurus construction [134, 135, 136], and edge-flow-based region segmentation [89]. The on-line demo is available at *http://vivaldi.ece.ucsb.edu/Netra/.*

2.3.7 MARS

MARS (multimedia analysis and retrieval system) was developed at University of Illinois at Urbana-Champaign [7, 22, 137, 49, 92, 61, 29, 138, 24, 23]. MARS differs from other systems in both its research scope and the techniques used. It is an interdisciplinary research effort involving multiple research communities: computer vision, database management system (DBMS), and information retrieval (IR). The research features of MARS are the integration of DBMS and IR (exact match with ranked retrieval) [7, 49], integration of indexing and retrieval (how the retrieval algorithm can take advantage of the underline indexing structure) [27], and integration of computer and human. The main focus of MARS is not on finding a single "best" feature representation, but rather on how to organize various visual features into a meaningful retrieval architecture that can dynamically adapt to different applications and different users. MARS formally proposes a relevance feedback architecture in image retrieval [23] and integrates such technique at various levels during retrieval, including query vector refinement [138], automatic matching tool selection [29], and and automatic feature adaption [24, 23]. The on-line demo is available at *http://jadzia.ifp.uiuc.edu:8000*.

2.3.8 Other systems

ART MUSEUM [139], developed in 1992, is one of the earliest content-based image retrieval systems. It uses edge feature as the visual feature for retrieval. Blobworld [140] developed at The University of California at Berkeley provides a transformation from the raw pixel data to a small set of localized coherent regions in color and texture space. This system allows the user to view the internal representation of the submitted image and the query results, thereby allowing the user to know why some "nonsimilar" images are returned. The user can then modify his or her query accordingly. The distinct feature of CAETIIML (*http://www.videolib.princeton.edu/test/retrieve*), built at Princeton University, is its combination of the on-line similarity searching and off-line subject searching [141]. More image retrieval systems can be found in [142, 143, 144, 145, 146, 147, 148, 149, 150].

CHAPTER 3

IMAGE ANALYSIS: VISUAL FEATURES USED IN THE THESIS

After discussing the general feature extraction techniques in the previous chapter, we will describe the image features used in this thesis. Specifically, the features include color, texture, and shape features. As discussed in the previous chapter, the effectiveness and robustness of feature extraction and representation is essential to good retrieval results.

3.1 Color Feature

To represent color, we choose the HSV space due to its de-correlated and uniform coordinates. We have tested two representations in our system: color histogram and color moments. For color histogram, since V coordinate is easily affected by the lighting condition, we use only HS coordinates to form an 8×8 two-dimensional histogram. To measure distance between two color histograms, we compute the amount of nonoverlap between the two histograms, which is defined as follows:

$$dist_{color} = 1 - \sum_{i=1}^{i=N} min(H_1(i), H_2(i)) \tag{3.1}$$

where H_1 and H_2 are the two histograms and N is the number of bins used in the histogram. The above intersection based measure of distance provides an accurate and efficient measure of (dis)similarity between two images based on their color.

For color moments, because high-order moments are independent of the intensity absolute values, we use all three channels in the HSV space. For each channel, the first three moments (mean, standard deviation, and skewness) are extracted. The similarity measure is generalized Euclidean.

3.2 Color Layout Feature

While the global color feature is useful for queries on the relative amount of each color in an image; it is not useful for queries on the spatial location of colors. For example, it is not possible to retrieve all images that contain a red region above and to the right of a large blue region based solely on the color histogram. Such queries can be answered correctly only if an image can be accurately segmented into regions of different color, which is difficult to achieve. But for queries relating to simple spatial relationships between colors, a relatively nonideal segmentation may still be sufficient.

To represent spatial arrangement of colors in an image, we do a simple k-means clustering on the HSV histogram of an image to produce a rough segmentation. For each region in the segmentation, we store the following information for color indexing: centroid, area, eccentricity, average color, and maximum bounding rectangle. Images will be searched by comparing the relative locations and colors of the indexed regions to see if they matched the color layout query.

3.3 Texture Feature

Texture feature is another very important feature in image retrieval. We have explored three texture representations.

3.3.1 Coarseness-contrast-directionality texture representation

To represent texture of an image, a CCD (coarseness, contrast, and directionality) texture feature representation was developed in [47, 48]. *Coarseness* is a measure of granularity of the

texture, i.e., fine versus coarse. *Contrast* represents the distribution of luminance of the image and is defined as

$$contrast = \sigma/(\alpha_4)^{1/4} \tag{3.2}$$

where $\alpha_4 = u_4/\sigma^4$. Here σ and u_4 are the standard deviation and the fourth central moment of the luminance, respectively. *Directionality* is a measure of how "directional" the image is. Using the above definitions of CCD, texture is represented as a set of three numbers. A problem with the above described CCD features is that it is sensitive to noise. We have developed and implemented an enhanced version of CCD by using histogram-based features. At each image pixel we compute these three texture features from the pixel's local neighborhood. The set of feature vectors from all image pixels forms the 3-D global texture histogram. We compute these measures for each image and use a weighted Euclidean distance function as the matching criteria. The method described by Tamura [47] (used by QBIC) uses three scalar measures, and does not consider the relationship between texture components. Our method includes this texture information and thus returns better matches (i.e., the textures are perceptually more similar).

3.3.2 Co-occurrence matrix texture representation

This approach explores the texture features by analyzing the gray-tone spatial dependencies [45]. We first define a matrix of relative frequencies with which two pixels separated by distance d at a specified angle occur on the image, one with gray-tone i and the other with gray-tone j. Such a matrix depends on the angle selected. We construct four such matrices, corresponding to 0, 45, 90 and 135 degrees, respectively.

After we construct the co-occurrence matrices, various statistical properties can be extracted as the texture feature. In this thesis, we use the most effective two features: contrast (CTR), and inverse difference moment (IDM).

3.3.3 Wavelet texture representation

An input image is fed into a wavelet filter bank and is decomposed into de-correlated subbands. Each subband captures the feature of some scale and orientation of the original image.

Specifically, we decompose an image into three wavelet levels; thus having 10 subbands. For each subband, the standard deviation of the wavelet coefficients is extracted. The 10 standard deviations are used as the texture representation for the image.

3.4 Shape Feature

Although shape is a very important feature that a human can easily extract from an image, reliable automatic extraction and representation of shapes is a challenging open problem in computer vision.

Some simple shape features are the perimeter, area, number of holes, eccentricity, symmetry, etc. Although these features easy to compute, they usually return too many false positives to be useful for content-based retrieval.

We use a modified Fourier descriptor (MFD) [61] in the MARS system. The proposed MFD satisfies the following four conditions:

1. Robustness to transformation: The representation must be invariant to translation, rotation, and scaling of shapes, as well as the starting point used in defining the boundary sequence.

2. Robustness to noise: Shape boundaries often contain local irregularities due to image noise. More importantly, spatial discretization introduces distortion along the entire boundary. The representation must be robust to these types of noise.

3. Feature extraction efficiency: Feature vectors should be computed efficiently.

4. Feature matching efficiency: Since matching is done on-line, the distance metric must require a *very* small computational cost.

3.5 Image Segmentation

Both the shape feature and the layout feature need the support from the image segmentation module. Our image segmentation is based on clustering and grouping in spatial-color-texture space. For a typical natural image, there is a high number of different colors and textures. A k-means clustering is one way to reduce the complexity while retaining salient color and texture features.

1. Randomly pick c starting points in the color-texture space as the initial means.
2. Cluster each point as belonging to the nearest neighbor mean.
3. Compute the new mean for each cluster.
4. Repeat 2 and 3 until all the clusters converge (i.e., when the number of pixels and mean value of each cluster do not change).

After this procedure, we have c clusters, each of which may corresponds to a set of image pixels. We define *cluster* as a natural group which has similar features of interest. The image pixels corresponding to a particular cluster may or *may not* be spatially contiguous. We define a *region* as one of the spatially connected regions corresponding to a cluster.

The k-means clustering generally produces regions of various sizes; some of the regions are very small (containing only a few pixels). We consider these regions as speckle noise and set a minimum region size threshold to filter out these small regions. The deleted regions are merged with the largest neighboring region.

After k-means clustering we have c clusters, each corresponding to several spatial regions. The next step is to extract the desired object from the regions.

One way to do this is to define a threshold in color-texture space. If a region's color-texture feature is above the threshold, then this region is considered the object. Otherwise, it is considered background. One obvious disadvantage of this thresholding method is that the threshold is image-dependent. We propose an attraction-based grouping method (ABGM) to overcome this disadvantage. The method is motivated by the way the human visual system might do the grouping.

As defined in physics,

$$F_{12} = G\frac{M_1 M_2}{d^2}$$

reflects how large the attraction is between the two masses M_1 and M_2 when they are of distance d. In ABGM, we use the similar concept, but now M_1 and M_2 are the size of the two regions, and d is the Euclidean distance between the two regions in six-dimensional (6-D) spatial-color-texture space.

The ABGM method is described as follows:

1. Choose attractor region A_i's from the clustered regions according to the knowledge of the application at hand.

2. Randomly choose an unlabeled region R_j. Find the attractions F_{ij} between A_i and R_j.

3. Associate region R_j with the attractor A_i that has the largest attraction to R_j.

4. Repeat steps 2 and 3 until all the regions are labeled.

5. Form the output segmentation by choosing the attractor of interest and its associated regions.

Note that if the attractor is bigger or closer (in 6-D space) to a unlabeled region, its attraction will be larger and thus the unlabeled region will be labeled to this attractor with higher probability. This is what a human visual system might do in the labeling process.

CHAPTER 4

IMAGE RETRIEVAL USING RELEVANCE FEEDBACK

Despite the extensive research effort, the retrieval techniques used in content-based image retrieval (CBIR) systems lag behind the corresponding techniques in today's best text search engines, such as Inquery [151], Alta Vista, and Lycos. One reason is that the information embedded in an image is far more complex than that in text. To better understand the history and methodology of CBIR and how we can improve CBIR's performance, we will first introduce an image object model before we go into the details of the discussions. An image object (O) can be modeled as a function of the image data (D), features (F), and representations (R). This is described below and also shown in Figure 4.1:

$$O = O(D, F, R) \tag{4.1}$$

- D is the raw image data, e.g., a JPEG image.
- $F = \{f_i\}, \quad i = 1, \cdots, I$ is a set of visual features associated with the image object, such as color, texture, and shape.
- $R_i = \{r_{ij}\}, \quad j = 1, \cdots, J_i$ is a set of representations for a given feature f_i , e.g., both color histogram and color moments are representations for the color feature [28]. Note that, each representation r_{ij} itself is normally a vector consisting of multiple components, i.e.,

  $$\vec{r}_{ij} = [r_{ij1}, ..., r_{ijk}, ..., r_{ijK_{ij}}] \tag{4.2}$$

  where K_{ij} is the length of the vector $\vec{r}_{ij}$.

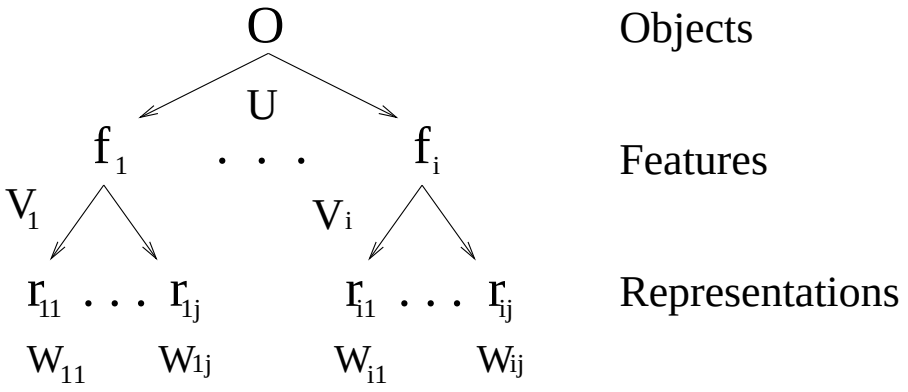


Figure 4.1 The image object model

This image model has three information abstraction levels (object, feature and representation), increasing in the information granularity. Furthermore, different weights (U at the object level, V_i at the feature level, and W_{ij} at the representation level) exist to reflect a particular entity's importance to its upper level. Most of the existing research in CBIR can fit in this comprehensive image object model, even though most research only explores part of it. We next briefly review the history and methodology of CBIR with respect to this model.

At the early stage of CBIR, research primarily focused on exploring what is the "best" feature (f_i) for a given image or the "best" representation (r_{ij}) for a given feature. For example, for the texture feature alone, almost a dozen representations have been proposed [20], including Tamura [48], MSAR [152], Word decomposition [120], Fractal [148], Gabor Filter [132, 9], and Wavelets [50, 51, 10]. Once the "best" features and representations are found, their corresponding weights (V_i and W_{ij}) are further fixed by the system designer based on his or her judgment of the importance of each r_{ij}'s and r_{ijk}'s. In addition, at the query stage, the user is asked to specify the weights at the object level (U). Based on the user's query image and specified weights, the retrieval system then tries to find similar images to the user's query.

In this type of approach, there is no human-computer interaction except that the user has to give a set of weights (U). We therefore refer to this type of approach as an *isolated* approach. While this approach establishes the basis of CBIR, its performance is not satisfactory because of the following two reasons:

- Specifying the object level weights (U) imposes a burden on the user, as it requires the user to have comprehensive knowledge of the visual features used in the retrieval system and how they related to his or her information need. This is not easy for an expert to do, let alone a normal user.

- Specifying feature and representation levels' weights (V_i and W_{ij}) imposes a burden on the system designer as well. Not knowing beforehand which r_{ij} and r_{ijk} can best match a particular user's perception of image content, it is almost impossible for the system designer to give accurate values for V_i and W_{ij}. Furthermore, these weights can not be easily obtained by training on typical users, because human perception of image content is subjective. That is, different persons, or the same person under different circumstances, may perceive the same visual content differently. This is called *human perception subjectivity* [23]. The subjectivity exists at various levels. For example, one person may be more interested in an image's color feature, and another may be more interested in the texture feature. Even if both people are interested in texture, how they perceive the similarity of texture may be quite different. This is illustrated in Figure 4.2.

(a) (b) (c)

Figure 4.2 Subjectivity in perceiving the texture feature (three textures in (a) to (c))

Among the three texture images, some may say that (a) and (b) are more similar if they do not care for the intensity contrast, while others may say that (a) and (c) are more similar if they ignore the local property on the seeds. No single texture representation (r_{ij}) can capture everything. Different representations capture the visual feature from different perspectives.

The above situation makes the *isolated* approach difficult to use. The burden imposed on both the user and system designer boils down to one thing. That is, in this approach, all the weights are *static* (fixed). For static weights, it is not possible to correctly model a user's information need.

Motivated by the limitations of the *isolated* approach, recent research in CBIR has moved to an interactive mechanism that involves a human as part of the retrieval process [20, 138, 113, 144]. Examples include *interactive* region segmentation [91], interactive image database annotation [113, 114], usage of *supervised* learning before the retrieval [134, 135], and *interactive* integration of keywords and high level concepts to enhance image retrieval performance [8, 153].

Based on the above analysis, in this chapter we explore *interactive* approaches that address the difficulties faced by the *isolated* approach. In particular, we will focus on techniques for interactive image retrieval based on *relevance feedback*, in which both user and computer interact to refine the user's true information need. Relevance feedback is a powerful technique first introduced in traditional text-based information retrieval systems. It is the process of automatically adjusting an existing query using the information fed back by the user about the relevance of previously retrieved objects such that the adjusted query is a better approximation to the user's information need [154, 26, 25]. In a *interactive* system, neither the user nor the system designer needs to specify any weights. The user only needs to mark which images he or she thinks are relevant to his or her query. The weights associated with the query object are *dynamically* updated to model the user's information need and perception subjectivity. In general, there are three approaches to relevance feedback in image retrieval. One is based on artificial intelligence (AI) learning techniques [113], one on Bayesian framework [155], and the other on information retrieval (IR) techniques [138, 156]. The last one is the most widely used and is the topic of this chapter.

The rest of the chapter is organized as follows. Section 4.1 describes a retrieval model based on the proposed image object model (Figure 4.1) and discusses how to compute the distance between two image objects. How to estimate weights U, V_i, and W_{ij}, as well as the query

vector, is essential to the *interactive* approach. We therefore explore various techniques in parameter (weights and queries) updating, ranging from heuristic-based approach to optimization approach. They are discussed in great detail in Sections 4.2-4.4. Experimental results of the proposed approaches over multiple data sets are given in Section 4.5. Concluding remarks are given in Section 4.6.

4.1 The Retrieval Process Based on Relevance Feedback

In order to compare the distance between two images, we need to define a retrieval model. The object model $O(D, F, R)$ together with a set of distance measures specifies a retrieval model.

Let the distance measures at the three levels be $\Phi(), \Theta()$ and $\Psi()$. How many arguments they have will depend on which forms (e.g., linear or quadratic) they are taking (see Equations (4.3), (4.30) and (4.51)). Let $\vec{r}_{mij},\ m = 1, \cdots, M;\ i = 1, \cdots, I;\ j = 1, \cdots, J_i$ be the the ij^{th} representation vector for the m^{th} image in the database, where M is the total number of images in the database, I the number of features, and J_i the number of representations for feature i. Let $\vec{q}_{ij},\ i = 1, \cdots, I;\ j = 1, \cdots, J_i$ be the query vector for the ij^{th} representation. The retrieval process can be described as follows and also illustrated in Figure 4.3.

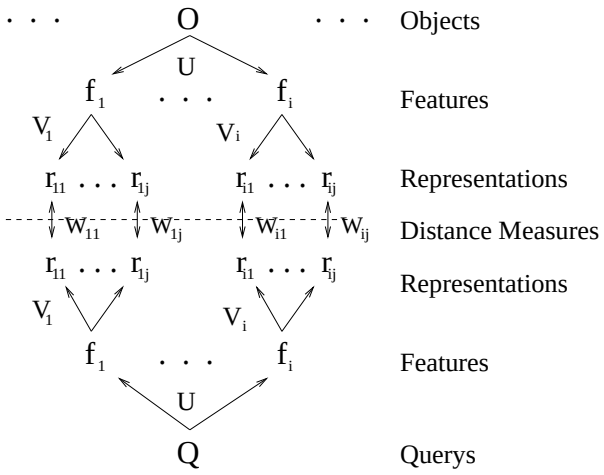


Figure 4.3 The retrieval process

1. Initialize the values of the weights U, V_i and W_{ij}.

2. The distance between an image and a query in terms of the ij^{th} representation is

$$\begin{aligned} d_m(\vec{r}_{ij}) &= \Psi_{ij}(\vec{r}_{mij}, \vec{q}_{ij}, W_{ij}) \\ m &= 1, \cdots, M \\ i &= 1, \cdots, I \\ j &= 1, \cdots, J_i \end{aligned}$$

   where $\vec{r}_{mij}$ is the ij^{th} representation vector for the m^{th} image in the database, and $d_m(\vec{r}_{ij})$ denotes the distance between the m^{th} image and a query in terms of representation ij.

3. The distance between the image and the query in terms of feature i is then

$$\begin{aligned} d_m(f_i) &= \Theta_i(\ d_m(\vec{r}_{ij}, V_i) \\ &= \Theta_i(\ \Psi_{ij}(\vec{r}_{mij}, \vec{q}_{ij}, W_{ij}), V_i) \end{aligned}$$

4. The overall distance is then

$$\begin{aligned} d_m &= \Phi(d_m(f_i, U) \\ &= \Phi(\Theta_i(\ \Psi_{ij}(\vec{r}_{mij}, \vec{q}_{ij}, W_{ij}), V_i), U) \end{aligned}$$

5. The images in the database are ordered by their overall distances to the query (d_m). The N_{RT} most similar ones are returned to the user, where N_{RT} is the number of images the user wants to retrieve.

6. For each of the retrieved images, the user provides a degree-of-relevance score, according to the user's information need and perception subjectivity.

7. The system dynamically updates the parameters (described in Sections 4.2-4.4) according to the user's feedback such that the adjusted query $\vec{q}_{ij}$ and weights U, V_i, W_{ij} better match the user's information need.

8. Go to Step 2 with the adjusted parameters and start a new iteration of retrieval until the user is satisfied.

In Figure 4.3, the information need embedded in the query Q flows up while the content of image object O flows down. They meet at the dashed line, where the distance measures are applied to calculate the distance values between Q and O.

Whether a retrieval model can update its parameters (weights or query vectors or both) distinguishes the *interactive* approach from the *isolated* approach. In the *isolated* approach, all the parameters (weights and query vectors) are fixed. Because of the fixed parameters, this approach cannot effectively model the user's information need and perception subjectivity. This approach also places a burden on both the user and the system designer. On the other hand, for the *interactive* approach, weights and query vectors are *dynamically* updated via relevance feedback. The burden of specifying the weights is removed from the user.

It is clear that the most essential part of the retrieval model is parameter updating. We next present two such approaches, a heuristic one, and an optimal one. The former is faster to compute but has less accuracy, and the latter is optimal for the given objective function but needs more time and computing power.

4.2 Parameter Updating: A Heuristic Approach

In this approach, we restrict the distance measures $\Phi()$ and $\Theta()_i$ to be linear functions in both their corresponding entities and their weights, and $\Psi_{ij}()$ to be linear in their weights but arbitrary in corresponding entities, that is,

$$\begin{aligned} d_m &= \Phi(\Theta_i\ (\ \Psi_{ij}\ (\vec{r}_{mij}, \vec{q}_{ij}, W_{ij}), V_i), U) \\ &= \sum_{i=1}^{I} u_i \sum_{j=1}^{J_i} v_{ij} \Psi_{ij}(\vec{r}_{mij}, \vec{q}_{ij}, \vec{w}_{ij}) \end{aligned}$$

where U takes the form of a vector $\vec{u}^T = [u_1, \cdots, u_i, \cdots, u_I]$ and u_i is the weight for the i^{th} feature; V_i takes the form of a vector $\vec{v}_i^T = [v_{i1}, \cdots, v_{ij}, \cdots, v_{iJ_i}]$ and v_{ij} is the weight for the

ij^{th} representation; W_{ij} takes the form of a vector $\vec{w}_{ij}^T = [w_{ij1}, \cdots, w_{ijk}, \cdots, w_{ijK_{ij}}]$ and w_{ijk} is the weight for the k^{th} component for representation ij; and T denotes transpose of a matrix or a vector.

The reason that we can assume $\Phi()$ and $\Theta_i()$ to be linear functions is that the weights are proportional to the entities' relative importance [157]. For example, if a user cares twice as much about one feature (color) as he does about another feature (shape), the overall similarity would be a linear combination of the two individual similarities with the weights being 2/3 and 1/3, respectively [157]. Note that, at the representation level, the distance measure Ψ_{ij} normally cannot be simplified to linear functions. For example, intersection distance is used in histogram comparison and Euclidean is used for wavelet texture comparison.

Because $\Phi()$ and $\Theta_i()$ are both linear functions, we can combine the object level and feature level into a single level, that is,

$$d_m = \sum_{i=1}^{I} u_i d_m(\vec{r}_i) \tag{4.3}$$

$$d_m(\vec{r}_i) = \Psi_i(\vec{r}_{mi}, \vec{q}_i, \vec{w}_i) \tag{4.4}$$

where I is now redefined as the total number of representations; i is the index for a particular representation; and u_i redefined as the weight directly from the image object to the i^{th} representation. We next discuss how to update these two levels' weights, as well as how to refine the query vector, in the following three subsections.

4.2.1 Update of the query vector $\vec{q}_i$

It is clear that the specification of the query vector $\vec{q}_i$ is critical, because the computed distance values (d_m's) are based on them. However, it is usually difficult for a user to map his or her information need into a query vector precisely. Relevance feedback is a way of refining the query vectors [26, 154, 25].

The mechanism of this method can be described elegantly in the vector space. If the sets of relevant images (D_R) and nonrelevant images (D_N) are known, the optimal query can be

proven to be [25, 26, 154]

$$\vec{q}_i^* = \frac{1}{N_R} \sum_{n \in D_R} \vec{x}_{ni} - \frac{1}{N_T - N_R} \sum_{i \in D_N} \vec{x}_{ni} \tag{4.5}$$

where N_R is the number of documents in D_R, N_T the number of the total documents, and $\vec{x}_{ni}$ is the n^{th} training sample in the sets D'_R and D'_N.

In practice, D_R and D_N are not known in advance. However, the relevance feedback obtained from the user furnishes approximations to D_R and D_N, which are referred as, D'_R and D'_N.

The original query $\vec{q}_i$ can be modified by putting more weights on the relevant components and less weights on the nonrelevant ones:

$$\begin{aligned} \vec{q}\,'_i &= \alpha \vec{q}_i + \beta(\frac{1}{N_{R'}} \sum_{n \in D'_R} \vec{x}_{ni}) - \gamma(\frac{1}{N_{N'}} \sum_{n \in D'_N} \vec{x}_{ni}) \\ i &= 1, \cdots, I \end{aligned}$$

where α, β, and γ are suitable constants [26, 154], and $N_{R'}$ and $N_{N'}$ are the numbers of objects in the relevant set D'_R and nonrelevant set D'_N.

4.2.2 Update of $\vec{w}_i$

The vector $\vec{w}_i$ consists of K_i components, associated with $\vec{r}_{mi}$ and $\vec{q}_i$. Each component w_{ik}, $k = 1, \cdots, K_i$ reflects the different contribution of a component to the representation vector $\vec{r}_i$. For example, in the wavelet texture representation, we know that the mean of a subband may be corrupted by the lighting condition, while the standard deviation of a subband is independent of the lighting condition. Therefore, more weight should be given to the standard deviation component, and less weight to the mean component. The support of different weights for $\vec{r}_i$'s enables the system to have more reliable feature representation and thus better retrieval performance.

A standard-deviation-based weight updating approach has been proposed in our previous work [138]. Out of the N_{RT} returned objects, for those objects that are marked with *highly relevant* or *relevant* by the user, stack their representation vector $\vec{r}_i$'s to form a $N \times K_i$ matrix, where N is the number of objects marked with *highly relevant* or *relevant*. In this way, each

column of the matrix is a length-N sequence of $r_{ik}, k = 1, \cdots, K_i$'s. Intuitively, if all the relevant objects have similar values for the component r_{ik}, it means that the component r_{ik} is a good indicator of the user's information need. On the other hand, if the values for the component r_{ik} are very different among the relevant objects, then r_{ik} is not a good indicator. Based on this analysis, the inverse of the standard deviation of the r_{ik} sequence is a good estimation of the weight w_{ik} for component r_{ik}. That is, the smaller the variance, the larger the weight and vice versa.

$$w_{ik} = \frac{1}{\sigma_{ik}} \tag{4.6}$$

where σ_{ik} is the standard deviation of the length-N sequence of r_{ik}'s. Here we assume that the user will mark more than two images as relevant or highly relevant, such that σ_{ik} is computable. Even though simple, we later show that this heuristic based weight updating approach is actually optimal under some conditions (Section 4.3.1).

4.2.3 Update of $\vec{u}$

The vector $\vec{u}$ consists of I elements, associated with each representation. Each u_i, $i = 1, \cdots, I$, reflects the user's different emphasis of a representation in the overall distance measure. The support of different weights enables the user to specify his or her information need more precisely. We next develop a heuristic approach to update u_i's according to the user's relevance feedback.

Let RT be the set of the most similar N_{RT} image objects according to the overall distance value d_m (Equation (4.3)):

$$RT = [RT_1, ..., RT_l, ..., RT_{N_{RT}}] \tag{4.7}$$

Let *Score* be the set containing the degree-of-relevance scores fed-back by the user for RT_l's (see Section 4.1)

$$= 3, \quad \text{if highly relevant} \tag{4.8}$$

$$= 1, \quad \text{if relevant} \tag{4.9}$$

$$Score_l = 0, \quad \text{if no-opinion} \tag{4.10}$$

$$= -1, \quad \text{if non-relevant} \tag{4.11}$$

$$= -3, \quad \text{if highly non-relevant} \tag{4.12}$$

The choice of 3, 1, 0, -1, and -3 as the scores (degree-of-relevance) is arbitrary. Experimentally we find that the above scores capture the semantic meaning of *highly relevant*, *relevant*, etc. In Equations (4.8)-(4.12), we provide the user with 5 levels of relevance. Although more levels may result in more accurate feedback, it is less convenient for the user to interact with the system. Experimentally we find that 5 levels is a good trade-off between convenience and accuracy.

For representation i, let RT^i be the set containing the most similar N_{RT} image objects to the query Q, according to the distance values $d_m(\vec{r}_i)$ (Equation (4.3)):

$$RT^i = [RT^i_1, ..., RT^i_l, ..., RT^i_{N_{RT}}] \tag{4.13}$$

To calculate the weight for representation i, first initialize $u_i = 0$, and then use the following procedure:

$$u_i = u_i + Score_l, \quad \text{if } RT^i_l \text{ is in } RT \tag{4.14}$$

$$= u_i + 0, \quad \text{if } RT^i_l \text{ is not in } RT \tag{4.15}$$

$$l = 0, ..., N_{RT} \tag{4.16}$$

Here, we consider all the images outside set RT as marked with *no opinion* and have the score of 0. After this procedure, if $u_i < 0$, set it to 0 (weights are non-negative). Intuitively, the above procedure means the more the overlap of relevant objects between RT and RT^i, the larger the weight of u_i. That is, if a representation reflects the user's information need, it receives more emphasis.

4.3 Parameter Updating: Optimal Approaches

In the previous section, we have discussed a heuristic approach to parameter updating, which is both easy to implement and quick to compute. However, because it is heuristic based, there is no optimality guarantee of any kind. In this and next sections we will explore optimization based approach for parameter updating. For simplicity, in the remaining discussion, we will combine the object level and feature level into a single level. That is, the distance between an image and a query is calculated as

$$d_m = \Phi(\ \Psi_i\ (\vec{r}_{mi}, \vec{q}_i, W_i), U) \tag{4.17}$$

where I is now redefined as the total number of representations; i is the index for a particular representation; and U is redefined as the weight directly from the image object to the i^{th} representation.

The remaining of the chapter is organized as follows. The parameter update for the representation level has attracted a lot of attention from researchers. The rest of this section is dedicated to this topic, where we review, analyze, and compare various approaches. How to update the parameters at the object level is much less addressed in the literature. To authors' best knowledge, the only seen approach is the one we discussed in Section 4.2. Even worse, there is no approach which can update both the object level and the representation level simultaneously. In the next section (Section 4.4), we will develop a few approaches to address this problem.

4.3.1 Parameter update at the representation level

Among the three levels in the image object model (Figure 4.1), the representation level is the most concrete, because the distance between an image and a query can be directly computed at this level. Because of this, researchers have developed many techniques for updating parameters (weights and query vectors) at this level.

The first paper appeared on this topic is [138]. In this paper, Rui et al. discussed two approaches for query refinement. One approach is to move the query vector

$$\begin{aligned} \vec{q}\,'_i &= \alpha\vec{q}_i + \beta(\frac{1}{N_{R'}}\sum_{n\in D'_R}\vec{x}_{ni}) - \gamma(\frac{1}{N_{N'}}\sum_{n\in D'_N}\vec{x}_{ni}) \\ i &= 1,\cdots,I \end{aligned}$$

where α, β, and γ are suitable constants [26, 154]; $N_{R'}$ and $N_{N'}$ are the numbers of objects in the relevant set D'_R and non-relevant set D'_N; and $\vec{x}_{ni}$ is the n^{th} training sample in the sets D'_R and D'_N.

The other approach is to update the weights associated with the query vector $\vec{q}$. As we have discussed in Section 4.2.2, a standard deviation approach was devised to compute w_{ik}:

$$w_{ik} = \frac{1}{\sigma_{ik}} \tag{4.18}$$

where σ_{ik} is the standard deviation of the length-N sequence of r_{ik}'s (see Section 4.2.2).

Since the appearance of the above approaches to relevance feedback in image retrieval, many improved versions have been proposed. One of the most elegant one is MindReader, developed by Ishikawa et al. [158]. After some revisions, the MindReader algorithm can be summarized as follows.

The distance function used for $\Psi_i()$ is a generalized Euclidean function. That is, the distance between an image and a query in terms of representation i is defined as:

$$d_m(r_i) = (\vec{r}_{mi} - \vec{q}_i)^T\ W_i\ (\vec{r}_{mi} - \vec{q}_i) \tag{4.19}$$

where T denotes transpose, and W_i is a $(K_i \times K_i)$ weight matrix associated with the i^{th} representation. Note that both $\vec{r}_{mi}$ and $\vec{q}_i$ are vectors of length K_i.

Let N denotes the number of training samples (the images fed back by the user). Let $\vec{\pi} = [\pi_1, \cdots, \pi_n, \cdots, \pi_N]$, $\pi_n > 0$ be the vector of degree-of-relevance for the corresponding training samples. They proposed to solve the following optimization problem

$$\begin{aligned} minJ &= \sum_{n=1}^{N}\ \pi_n\ (\vec{x}_{ni} - \vec{q}_i)^T\ W_i\ (\vec{x}_{ni} - \vec{q}_i) \\ s.t. &\quad det(W_i) = 1 \end{aligned}$$

where $\vec{x}_{ni}$ denotes the n^{th} training sample and is a vector of length K_i.

By forming the Lagrange multiplier as

$$L = J + \lambda \left(det(W_i) - 1 \right) \tag{4.20}$$

we can obtain the optimal solutions for $\vec{q}_i$ and W_i [158]

$$\vec{q}_i{}^{T^*} = \frac{\vec{\pi}^{\,T} X_i}{\sum_{n=1}^{N} \pi_n} \tag{4.21}$$

$$W_i^* = (det(C_i))^{\frac{1}{J_i}} C_i^{-1} \tag{4.22}$$

where X_i is the training sample matrix for representation i, obtained by stacking the N training vectors ($\vec{x}_{ni}$) into a matrix. It is therefore a $(N \times K_i)$ matrix. The term C_i is the $(K_i \times K_i)$ weighted covariance matrix of X_i. That is,

$$C_{i_{rs}} = \frac{\sum_{n=1}^{N} \pi_n \left(x_{nr} - q_r \right) \left(x_{ns} - q_s \right)}{\sum_{n=1}^{N} \pi_n}$$

$$r, s = 1, \cdots, K_i$$

Both solutions match our intuition nicely. The optimal query vector is the weighted average of the training samples. The optimal weight matrix is inversely proportional to the covariance matrix of the training samples.

MindReader formulated an optimization problem to solve for best $\vec{q}_i$ and W_i. We can arrive at similar and even simpler solutions by formulating a MLE (maximum likelihood estimation) problem. Assume the training samples obey the normal distribution of $N(\vec{\mu}_i, \Sigma_i)$, where $\vec{\mu}_i$ and Σ_i are the mean vector and covariance matrix respectively, and they are independent and identically distributed (iid). The training samples therefore will form a single cloud or cluster and the centroid of the cluster is the ideal query vector, which achieves minimum distance distortion and maximum likelihood. That is, $\vec{q}_i = \vec{\mu}_i$. We further define the distance between a training sample and the query vector to be the *Mahalanobis distance*:

$$d_n(r_i) = (\vec{x}_{ni} - \vec{\mu}_i)^T \Sigma_i^{-1} (\vec{x}_{ni} - \vec{\mu}_i) \tag{4.23}$$

where Σ_i^{-1} plays the role of W_i.

Assume the training samples are iid, the likelihood function and log likelihood function can be written as

$$\begin{aligned} L(\vec{\mu}_i, \Sigma_i) &= \prod_{n=1}^{N} p(\vec{x}_{ni}|\vec{\mu}_i, \Sigma_i) \\ LL(\vec{\mu}_i, \Sigma_i) &= \sum_{n=1}^{N} p(\vec{x}_{ni}|\vec{\mu}_i, \Sigma_i) \end{aligned}$$

The degree-of-relevance vector $\vec{\pi}$ comes into play as the relative frequency of a particular training sample. That is, the higher its degree-of-relevance, the more frequently the training sample will appear in the training set. By taking this into account, the log likelihood function can be written as:

$$LL(\vec{\mu}_i, \Sigma_i) = \sum_{n=1}^{N} \pi_n \; p(\vec{x}_{ni}|\vec{\mu}_i, \Sigma_i) \tag{4.24}$$

It is easy to prove the MLE for $\vec{\mu}_i, \Sigma_i$ is [98]:

$$\vec{q}_i^{\,T^*} = \vec{\mu}_i^{\,T^*} = \frac{\vec{\pi}^{\,T} X_i}{\sum_{n=1}^{N} \pi_n} \tag{4.25}$$

$$W_i^* = \Sigma_i^{-1^*} = C_i^{-1} \tag{4.26}$$

Comparing Equations (4.21)-(4.22) and (4.25)-(4.26), we can see that from different ways, we have arrived at similar solutions. The physical meanings of Equations (4.25) and (4.26) are even more intuitive and satisfactory.

After deriving the optimal solutions for W_i, if we look back at the heuristic approach (Equation (4.6)) proposed in [138], it is actually optimal under some assumptions. That is, if we restrict W_i to be a diagonal matrix (i.e., we use Euclidean distance rather than generalized Euclidean distance), the solution in [138] is optimal. This is very obvious, since from both Equations (4.22) and (4.26) the diagonal weights are inverse proportional to the standard deviation.

4.3.2 Practical considerations

We have shown the optimality of the solutions in the previous subsection. In this subsection, we will further discuss some real-world issues.

In both solutions for W_i (Equations (4.22) and (4.26)), we need to compute the inverse of the covariance matrix C_i. It is clear that, if $N < K_i$, C_i is not invertible, and we thus cannot have W_i. In MindReader, the authors proposed a solution to solve this by using a pseudo-inverse defined below.

The singular value decomposition (SVD) of C_i is

$$C_i = A\ \Lambda\ B^T \tag{4.27}$$

where Λ is a diagonal matrix: $diag(\lambda_1, \cdots, \lambda_j, \cdots, \lambda_{K_i})$. Those λ's are either positive or zero. Suppose there are L nonzero λ's, the pseudo-inverse of C_i is defined as

$$\begin{aligned} C_i^+ &= A\ \Lambda^+\ B^T \\ \Lambda^+ &= diag(\frac{1}{\lambda_1}, \cdots, \frac{1}{\lambda_L}, 0, \cdots, 0). \end{aligned}$$

where $+$ denotes the pseudo-inverse of a matrix. The approximation solution of W_i^* is then [158]

$$W_i^* = (\prod_{l=1}^{L}\ \lambda_l)^{\frac{1}{L}}\ C_i^+ \tag{4.28}$$

Even though, in theory, we can get around the *singular* problem by using the above procedure, in reality this solution does not give satisfactory results. This is especially true when N is far less then K_i. Remember, we need to use $(N-1) \times K_i$ numbers from the training samples to estimate $\frac{K_i\ (K_i+1)}{2}$ parameters in C_i. In MindReader, the authors used a $K_i = 2$ example to show the performance of the algorithm. However, in real image retrieval systems, $K_i = 2$ is not very realistic. For example, in HSV color histogram, an $(8 \times 4 \times 4 = 32)$ vector is normally used. In other shape and texture representations, we also have high-dimensionality vectors [20].

Considering this, in practice it is not bad to use the standard deviation approach (Equation (4.6)), especially when $N < K_i$. This approach is much more robust, as it requires only two different training samples to estimate the parameters.

4.4 Parameter Update at Both Levels: Optimal Approaches

As we can see from the previous section, there are many approaches to parameter updating at the representation level. Unfortunately, there is almost no research in optimizing both levels simultaneously.

One natural thought from previous section is to stack all the representation vectors $\vec{r_i}$ to form a huge overall vector $\vec{r}$ and directly use the results from previous section. In theory, this will work, as long as we have enough training samples. Unfortunately, as we discussed in Section 4.3.2, even for each individual $\vec{r_i}$ we have the risk of not being able to invert C_i, let alone to deal with the stacked huge vector $\vec{r}$. This suggests us that using a flat model (stacking all $\vec{r_i}$'s together) will not work in practice. We therefore need to explore approaches that can take advantage of the hierarchical image object model (Figure 4.1). Recall that the overall distance between a training sample and a query is:

$$d_n = \Phi(\ \Psi_i\ (\vec{x}_{ni}, \vec{q}_i, W_i), U) \tag{4.29}$$

The goal of parameter updating is to minimize the weighted distance between all the training samples and the query. Because Φ and Ψ_i can be nonlinear functions in general, a direct optimization of J over $U, W_i, \vec{q}_i$ is intractable. Depending on which distance functions we use, there are different algorithms. We next examine two of them.

4.4.1 Quadratic in both $\Phi()$ and $\Psi_i()$

In this subsection, we restrict ourselves to use the generalized Euclidean distance for both $\Phi()$ and $\Psi_i()$. That is,

$$d_n = \vec{g}_n^T\ U\ \vec{g}_n \tag{4.30}$$

$$\vec{g}_n = [g_{n1}, \cdots, g_{ni}, \cdots, g_{nI}] \tag{4.31}$$

$$g_{ni} = (\vec{x}_{ni} - \vec{q}_i)\ ^T W_i (\vec{x}_{ni} - \vec{q}_i) \tag{4.32}$$

The parameter updating process can therefore be formulated as a constrained optimization problem:

$$
\begin{aligned}
min\ J &= \vec{\pi}^{\ T} \times \vec{d} && (4.33)\\
d_n &= \vec{g}_n^{T}\ U\ \vec{g}_n && (4.34)\\
\vec{g}_n &= [g_{n1}, \cdots, g_{ni}, \cdots, g_{nI}] && (4.35)\\
g_{ni} &= (\vec{x}_{ni} - \vec{q}_i)^{\ T} W_i (\vec{x}_{ni} - \vec{q}_i) && (4.36)\\
s.t. &\quad det(U) = 1 && (4.37)\\
&\quad det(W_i) = 1 && (4.38)\\
n &= 1, \cdots, N && (4.39)\\
i &= 1, \cdots, I && (4.40)
\end{aligned}
$$

We use Lagrange multipliers to solve this constrained optimization problem:

$$
L = \vec{\pi}^{\ T} \times \vec{d} - \lambda(det(U) - 1) - \sum_{i=1}^{I} \lambda_i (det(W_i) - 1) \qquad (4.41)
$$

4.4.1.1 Optimal solution for $\vec{q}_i$

$$
\begin{aligned}
\frac{\partial L}{\partial \vec{q}_i} &= \vec{\pi}^{\ T} \times \begin{bmatrix} \frac{\partial d_1}{\partial \vec{q}_i} \\ \cdots \\ \frac{\partial d_n}{\partial \vec{q}_i} \\ \cdots \\ \frac{\partial d_N}{\partial \vec{q}_i} \end{bmatrix} \\
&= \vec{\pi}^{\ T} \times \begin{bmatrix} 2\vec{g}_1^{T}\ U\ \frac{\partial \vec{g}_1}{\partial \vec{q}_i} \\ \cdots \\ 2\vec{g}_n^{T}\ U\ \frac{\partial \vec{g}_n}{\partial \vec{q}_i} \\ \cdots \\ 2\vec{g}_N^{T}\ U\ \frac{\partial \vec{g}_N}{\partial \vec{q}_i} \end{bmatrix}
\end{aligned}
$$

$$
= \quad \vec{\pi}^{\,T} \times \left[\begin{array}{ccc}
 & & \left[\begin{array}{c} \frac{\partial g_{11}}{\partial \vec{q}_i} \\ \cdots \\ \frac{\partial g_{1i}}{\partial \vec{q}_i} \\ \cdots \\ \frac{\partial g_{1I}}{\partial \vec{q}_i} \end{array} \right] \\
2\vec{g}_1^{\,T}\ U & \times & \cdots \\
 & & \left[\begin{array}{c} \frac{\partial g_{n1}}{\partial \vec{q}_i} \\ \cdots \\ \frac{\partial g_{ni}}{\partial \vec{q}_i} \\ \cdots \\ \frac{\partial g_{nI}}{\partial \vec{q}_i} \end{array} \right] \\
2\vec{g}_n^{\,T}\ U & \times & \cdots \\
 & & \left[\begin{array}{c} \frac{\partial g_{N1}}{\partial \vec{q}_i} \\ \cdots \\ \frac{\partial g_{Ni}}{\partial \vec{q}_i} \\ \cdots \\ \frac{\partial g_{NI}}{\partial \vec{q}_i} \end{array} \right] \\
2\vec{g}_N^{\,T}\ U & \times &
\end{array} \right]
$$

$$
= \quad \vec{\pi}^{\,T} \times \begin{bmatrix}
-4\ \vec{g}_1^{\,T}\ U\ \times \begin{bmatrix} 0 \\ \cdots \\ (\vec{x}_{1i} - \vec{q}_i)^{\,T}\ W_i \\ \cdots \\ 0 \end{bmatrix} \\
\cdots \\
-4\ \vec{g}_n^{\,T}\ U\ \times \begin{bmatrix} 0 \\ \cdots \\ (\vec{x}_{ni} - \vec{q}_i)^{\,T}\ W_i \\ \cdots \\ 0 \end{bmatrix} \\
\cdots \\
-4\ \vec{g}_N^{\,T}\ U\ \times \begin{bmatrix} 0 \\ \cdots \\ (\vec{x}_{Ni} - \vec{q}_i)^{\,T}\ W_i \\ \cdots \\ 0 \end{bmatrix}
\end{bmatrix}
$$

$$
\begin{aligned}
&= \vec{\pi}^{\,T} \times \begin{bmatrix}
-4\ \vec{h}_1^T \times \begin{bmatrix} 0 \\ \cdots \\ (\vec{x}_{1i} - \vec{q}_i)^{\,T}\ W_i \\ \cdots \\ 0 \end{bmatrix} \\
\cdots \\
-4\ \vec{h}_n^T \times \begin{bmatrix} 0 \\ \cdots \\ (\vec{x}_{ni} - \vec{q}_i)^{\,T}\ W_i \\ \cdots \\ 0 \end{bmatrix} \\
\cdots \\
-4\ \vec{h}_N^T \times \begin{bmatrix} 0 \\ \cdots \\ (\vec{x}_{Ni} - \vec{q}_i)^{\,T}\ W_i \\ \cdots \\ 0 \end{bmatrix}
\end{bmatrix} \\
&= \vec{\pi}^{\,T} \times \begin{bmatrix}
-4\ h_{1i}^T \times (\vec{x}_{1i} - \vec{q}_i)^{\,T}\ W_i \\
\cdots \\
-4\ h_{ni}^T \times (\vec{x}_{ni} - \vec{q}_i)^{\,T}\ W_i \\
\cdots \\
-4\ h_{Ni}^T \times (\vec{x}_{Ni} - \vec{q}_i)^{\,T}\ W_i
\end{bmatrix}
\end{aligned}
$$

where $\vec{h}_n^T = \vec{g}_n^T\ U$, and h_{ni} is the ith component of $\vec{h}_n$.

Setting the above equation to zero, we have

$$\vec{\pi}^{\ T} \times \begin{bmatrix} -4\ h_{1i}^T \times (\vec{x}_{1i} - \vec{q}_i)^{\ T}\ W_i \\ \cdots \\ -4\ h_{ni}^T \times (\vec{x}_{ni} - \vec{q}_i)^{\ T}\ W_i \\ \cdots \\ -4\ h_{Ni}^T \times (\vec{x}_{Ni} - \vec{q}_i)^{\ T}\ W_i \end{bmatrix} = 0$$

$$\vec{\pi}^{\ T} \times \begin{bmatrix} h_{1i}^T \times (\vec{x}_{1i} - \vec{q}_i)^{\ T} \\ \cdots \\ h_{ni}^T \times (\vec{x}_{ni} - \vec{q}_i)^{\ T} \\ \cdots \\ h_{Ni}^T \times (\vec{x}_{Ni} - \vec{q}_i)^{\ T} \end{bmatrix} = 0$$

$$\vec{\tilde{\pi}}^{\ T} \times \begin{bmatrix} (\vec{x}_{1i} - \vec{q}_i)^{\ T} \\ \cdots \\ (\vec{x}_{ni} - \vec{q}_i)^{\ T} \\ \cdots \\ (\vec{x}_{Ni} - \vec{q}_i)^{\ T} \end{bmatrix} = 0$$

where $\tilde{\pi}_{ni} = \pi_n \times h_{ni}$. Note also that we have used the fact that W_i is invertible since $det(W_i) = 1$. The final solution of $\vec{q}_i$ is:

$$\vec{q}_i^{\ T^*} = \frac{\vec{\tilde{\pi}}^{\ T}\ X_i}{\sum_{n=1}^{N} \tilde{\pi}_{ni}} \tag{4.42}$$

4.4.1.2 Optimal solution for W_i

Before we explore how to find the optimal solution of W_i, we note that the constraint $det(W_i) = 1$ can be rewritten as

$$\begin{aligned} \sum_{r=1}^{K_i} (-1)^{r+s} w_{i_{rs}} det(W_{i_{rs}}) &= 1 \\ r &= 1, \cdots, K_i \\ s &= 1, \cdots, K_i \end{aligned}$$

where $det(W_{i_{rs}})$ is the $(rs)^{th}$ minor of W_i, and $W_i = [w_{i_{rs}}]$.

This equation can be further rewritten as [158]:

$$\sum_{r=1}^{K_i} \sum_{s=1}^{K_i} (-1)^{r+s} w_{i_{rs}} det(W_{i_{rs}}) = K_i \tag{4.43}$$

To obtain the optimal solution of W_i, we take partial derivative of L with respect to $w_{i_{rs}}, r, s = 1, \cdots, K_i$:

$$\frac{\partial L}{\partial w_{i_{rs}}} = \vec{\pi}^{\,T} \times \begin{bmatrix} 2\vec{g}_1^T \; U \; \frac{\partial \vec{g}_1}{\partial w_{i_{rs}}} \\ \cdots \\ 2\vec{g}_n^T \; U \; \frac{\partial \vec{g}_n}{\partial w_{i_{rs}}} \\ \cdots \\ 2\vec{g}_N^T \; U \; \frac{\partial \vec{g}_N}{\partial w_{i_{rs}}} \end{bmatrix} - \lambda_i \; (-1)^{r+s} det(W_{i_{rs}})$$

$$
\begin{aligned}
&= \vec{\pi}^{\,T} \times \begin{bmatrix}
2\ \vec{g}_1^T\ U\ \times \begin{bmatrix} 0 \\ \cdots \\ \frac{\partial q_{1i}}{\partial w_{i_{rs}}} \\ \cdots \\ 0 \end{bmatrix} \\
\cdots \\
2\ \vec{g}_n^T\ U\ \times \begin{bmatrix} 0 \\ \cdots \\ \frac{\partial q_{ni}}{\partial w_{i_{rs}}} \\ \cdots \\ 0 \end{bmatrix} \\
\cdots \\
2\ \vec{g}_N^T\ U\ \times \begin{bmatrix} 0 \\ \cdots \\ \frac{\partial q_{Ni}}{\partial w_{i_{rs}}} \\ \cdots \\ 0 \end{bmatrix}
\end{bmatrix} - \lambda_i\ (-1)^{r+s} det(W_{i_{rs}}) \\
&= \vec{\pi}^{\,T} \times \begin{bmatrix}
2\ h_{1i}^T \times (x_{1ir} - q_{ir})(x_{1is} - q_{is}) \\
\cdots \\
2\ h_{ni}^T \times (x_{nir} - q_{ir})(x_{1is} - q_{is}) \\
\cdots \\
2\ h_{Ni}^T \times (x_{Nir} - q_{ir})(x_{1is} - q_{is})
\end{bmatrix} - \lambda_i\ (-1)^{r+s} det(W_{i_{rs}}) \\
&= 2\ \sum_{n=1}^{N} \tilde{\pi}_n\ (x_{nir} - q_{ir})(x_{1is} - q_{is}) - \lambda_i\ (-1)^{r+s} det(W_{i_{rs}})
\end{aligned}
$$

Set the above equation to zero, we will get:

$$
2\ \sum_{n=1}^{N} \tilde{\pi}_n\ (x_{nir} - q_{ir})(x_{1is} - q_{is}) - \lambda_i\ (-1)^{r+s} det(W_{i_{rs}}) = 0
$$

$$2 \sum_{n=1}^{N} \tilde{\pi}_n \ (x_{nir} - q_{ir})(x_{1is} - q_{is}) = \lambda_i \ (-1)^{r+s} det(W_{i_{rs}})$$

$$det(W_{i_{rs}}) = \frac{2 \ \sum_{n=1}^{N} \tilde{\pi}_n \ (x_{nir} - q_{ir})(x_{1is} - q_{is})}{\lambda_i \ (-1)^{r+s}}$$

Define $w_{i_{rs}}^{-1}$ to be the $(rs)^{th}$ element of matrix W_i^{-1}. We then have

$$\begin{aligned}
w_{i_{rs}}^{-1} &= \frac{(-1)^{r+s} \ det(W_{i_{rs}})}{det(W_i)} \\
&= (-1)^{r+s} \ det(W_{i_{rs}}) \\
&= (-1)^{r+s} \ \frac{2 \ \sum_{n=1}^{N} \tilde{\pi}_n \ (x_{nir} - q_{ir})(x_{1is} - q_{is})}{\lambda_i \ (-1)^{r+s}} \\
&= \frac{2 \ \sum_{n=1}^{N} \tilde{\pi}_n \ (x_{nir} - q_{ir})(x_{1is} - q_{is})}{\lambda_i} \\
&= 2 \sum_{n=1}^{N} \tilde{\pi}_n \frac{\sum_{n=1}^{N} \tilde{\pi}_n \ (x_{nir} - q_{ir})(x_{1is} - q_{is})}{\lambda_i \sum_{n=1}^{N} \tilde{\pi}_n} \\
&= \frac{1}{\lambda_i'} \frac{\sum_{n=1}^{N} \tilde{\pi}_n \ (x_{nir} - q_{ir})(x_{1is} - q_{is})}{\sum_{n=1}^{N} \tilde{\pi}_n}
\end{aligned}$$

where $\lambda_i' = \lambda_i / 2 \sum_{n=1}^{N} \tilde{\pi}_n$.

Knowing $\vec{q}_i$ is just the weighted average of $\vec{x}_{ni}$'s, it is easy to observe that $\frac{\sum_{n=1}^{N} \tilde{\pi}_n (x_{nis} - q_{is})(x_{1it} - q_{it})}{\sum_{n=1}^{N} \tilde{\pi}_n}$ is nothing but the $(r,s)^{th}$ element of the weighted covariance matrix (C_i) of X_i. That is, $C_i = \lambda_i' \ W_i^{-1}$. Take determinant of both sides, we have:

$$\begin{aligned}
\lambda_i'^{K_i} det(W_i^{-1}) &= det(C_i) \\
\lambda_i'^{K_i} &= det(C_i) \\
\lambda_i' &= det(C_i)^{\frac{1}{K_i}}
\end{aligned}$$

Therefore the optimal solution for W_i is:

$$W_i^* = (det(C_i))^{\frac{1}{K_i}} \ C_i^{-1} \tag{4.44}$$

4.4.1.3 Optimal solution for U

$$\begin{aligned}\frac{\partial L}{\partial u_{rs}} &= \vec{\pi}^{\,T} \times \begin{bmatrix} g_{1r}\ g_{1s} \\ \cdots \\ g_{nr}\ g_{ns} \\ \cdots \\ g_{Nr}\ g_{Ns} \end{bmatrix} - \lambda\ (-1)^{r+s} det(U_{rs}) \\ &= \sum_{n=1}^{N} \pi_n\ g_{nr}\ g_{ns} - \lambda(-1)^{r+s} det(U_{rs})\end{aligned}$$

where $det(U_{rs})$ is the $(rs)^{th}$ minor of U.

Set the above equation to zero, we will get:

$$\begin{aligned}\sum_{n=1}^{N} \pi_n\ g_{nr}\ g_{ns} - \lambda(-1)^{r+s} det(U_{rs}) &= 0 \\ \sum_{n=1}^{N} \pi_n\ g_{nr}\ g_{ns} &= \lambda(-1)^{r+s} det(U_{rs})\end{aligned}$$

This equation has a very similar form to Equation (4.44). We know its solution is:

$$U^* = (det(R))^{\frac{1}{I}}\ R^{-1} \tag{4.45}$$

where R is the weighted correlation matrix of $\vec{g}_n, n = 1, \cdots, N$.

$$\begin{aligned}R_{rs} &= \sum_{n=1}^{N} \pi_n\ g_{nr}\ g_{ns} \\ r &= 1, \cdots, I \\ s &= 1, \cdots, I\end{aligned}$$

4.4.1.4 Convergence and performance

Even though we have obtained explicit optimal solutions

$$\vec{q}_i^{\ T^*} = \frac{\vec{\tilde{\pi}}^{\,T}\ X_i}{\sum_{n=1}^{N} \tilde{\pi}_i} \tag{4.46}$$

$$W_i^* = (2\sum_{n=1}^{N} \tilde{\pi}_n det(C_i))^{\frac{1}{K_i}} \; C_i^{-1} \tag{4.47}$$

$$U^* = (det(R))^{\frac{1}{I}} \; R^{-1} \tag{4.48}$$

they actually depend on

$$g_{ni} = (\vec{x}_{ni} - \vec{q}_i)^{\,T} W_i (\vec{x}_{ni} - \vec{q}_i) \tag{4.49}$$

which in turn depends on $\vec{q}_i$ and W_i.

To prove its convergence is not easy, since J is highly nonlinear in $\vec{q}_i$ and W_i. If indeed we can prove its convergence, it is at most an iterative algorithm. In theory this works fine, but for a particular application such as image retrieval, this is not desirable. Fast response time is always one of the most important requirements in image retrieval. The above discussion shows that if both $\Phi()$ and $\Psi_i()$ take quadratic form, the algorithm will not work in practice. We then need to explore other suitable forms for $\Phi()$ and $\Psi_i()$.

4.4.2 Linear in $\Phi()$ and quadratic in $\Psi_i()$

Because there is no explicit solution for $\vec{q}_i$, W_i, and U when both $\Phi()$ and $\Psi()$ are quadratic, we further simplify $\Phi()$ to be a linear function:

$$min \; J = \vec{\pi}^{\,T} \times \vec{d} \tag{4.50}$$

$$d_n = \vec{u}^T \vec{g}_n \tag{4.51}$$

$$\vec{g}_n = [g_{n1}, \cdots, g_{ni}, \cdots, g_{nI}] \tag{4.52}$$

$$g_{ni} = (\vec{x}_{ni} - \vec{q}_i)^{\,T} W_i (\vec{x}_{ni} - \vec{q}_i) \tag{4.53}$$

$$s.t. \quad \sum_{i=1}^{I} \frac{1}{u_i} = 1 \tag{4.54}$$

$$det(W_i) = 1 \tag{4.55}$$

$$n = 1, \cdots, N \tag{4.56}$$

$$i = 1, \cdots, I \tag{4.57}$$

where the matrix U now reduces to a vector $\vec{u}$ and its constraint is changed accordingly as shown in Equation (4.54).

Again, we use Lagrange multipliers to solve this constrained optimization problem:

$$L = \vec{\pi}^{\,T} \times \vec{d} - \lambda(\sum_{i=1}^{I} \frac{1}{u_i} - 1) - \sum_{i=1}^{I} \lambda_i (det(W_i) - 1) \tag{4.58}$$

4.4.2.1 Optimal solution for $\vec{q}_i$

$$\begin{aligned} \frac{\partial L}{\partial \vec{q}_i} &= \vec{\pi}^{\,T} \times \begin{bmatrix} \frac{\partial d_1}{\partial \vec{q}_i} \\ \cdots \\ \frac{\partial d_n}{\partial \vec{q}_i} \\ \cdots \\ \frac{\partial d_N}{\partial \vec{q}_i} \end{bmatrix} \\ &= \vec{\pi}^{\,T} \times \begin{bmatrix} \vec{u}^T \ \frac{\partial \vec{q}_1}{\partial \vec{q}_i} \\ \cdots \\ \vec{u}^T \ \frac{\partial \vec{q}_n}{\partial \vec{q}_i} \\ \cdots \\ \vec{u}^T \ \frac{\partial \vec{q}_N}{\partial \vec{q}_i} \end{bmatrix} \end{aligned}$$

$$
= \quad \vec{\pi}^{\,T} \times \left[\begin{array}{l}
\vec{u}^T \quad \times \left[\begin{array}{c} \frac{\partial g_{11}}{\partial \vec{q}_i} \\ \cdots \\ \frac{\partial g_{1i}}{\partial \vec{q}_i} \\ \cdots \\ \frac{\partial g_{1I}}{\partial \vec{q}_i} \end{array} \right] \\
\cdots \\
\vec{u}^T \quad \times \left[\begin{array}{c} \frac{\partial g_{n1}}{\partial \vec{q}_i} \\ \cdots \\ \frac{\partial g_{ni}}{\partial \vec{q}_i} \\ \cdots \\ \frac{\partial g_{nI}}{\partial \vec{q}_i} \end{array} \right] \\
\cdots \\
\vec{u}^T \quad \times \left[\begin{array}{c} \frac{\partial g_{N1}}{\partial \vec{q}_i} \\ \cdots \\ \frac{\partial g_{Ni}}{\partial \vec{q}_i} \\ \cdots \\ \frac{\partial g_{NI}}{\partial \vec{q}_i} \end{array} \right]
\end{array} \right]
$$

$$
\begin{aligned}
&= \vec{\pi}^{\,T} \times \begin{bmatrix}
-2\ \vec{u}^T \ \times \begin{bmatrix} 0 \\ \cdots \\ (\vec{x}_{1i} - \vec{q}_i)^{\,T}\ W_i \\ \cdots \\ 0 \end{bmatrix} \\
\cdots \\
-2\ \vec{u}^T \ \times \begin{bmatrix} 0 \\ \cdots \\ (\vec{x}_{ni} - \vec{q}_i)^{\,T}\ W_i \\ \cdots \\ 0 \end{bmatrix} \\
\cdots \\
-2\ \vec{u}^T \ \times \begin{bmatrix} 0 \\ \cdots \\ (\vec{x}_{Ni} - \vec{q}_i)^{\,T}\ W_i \\ \cdots \\ 0 \end{bmatrix}
\end{bmatrix} \\
&= \vec{\pi}^{\,T} \times \begin{bmatrix}
-2\ u_i\ (\vec{x}_{1i} - \vec{q}_i)^{\,T}\ W_i \\
\cdots \\
-2\ u_i\ (\vec{x}_{ni} - \vec{q}_i)^{\,T}\ W_i \\
\cdots \\
-2\ u_i\ (\vec{x}_{Ni} - \vec{q}_i)^{\,T}\ W_i
\end{bmatrix}
\end{aligned}
$$

By setting the above equation to zero, we will have:

$$\vec{\pi}^{\,T} \times \begin{bmatrix} -2\ u_i\ (\vec{x}_{1i} - \vec{q}_i)^{\,T}\ W_i \\ \cdots \\ -2\ u_i\ (\vec{x}_{ni} - \vec{q}_i)^{\,T}\ W_i \\ \cdots \\ -2\ u_i\ (\vec{x}_{Ni} - \vec{q}_i)^{\,T}\ W_i \end{bmatrix} = 0$$

$$\vec{\pi}^{\,T} \times \begin{bmatrix} (\vec{x}_{1i} - \vec{q}_i)^{\,T} \\ \cdots \\ (\vec{x}_{ni} - \vec{q}_i)^{\,T} \\ \cdots \\ (\vec{x}_{Ni} - \vec{q}_i)^{\,T} \end{bmatrix} = 0$$

The final solution of $\vec{q}_i$ is:

$$\vec{q}_i^{\,T^*} = \frac{\vec{\pi}^{\,T}\ X_i}{\sum_{n=1}^{N} \pi_n} \tag{4.59}$$

This solution closely matches our intuition. That is, $\vec{q}_i^{T^*}$ is nothing but the weighted average of the training samples. It is also exactly the same as the optimal solutions obtained in MindReader (Equation (4.21)) and MLE (Equation (4.25)).

4.4.2.2 Optimal solution for W_i

$$\frac{\partial L}{\partial w_{i_{rs}}} = \vec{\pi}^{\,T} \times \begin{bmatrix} \vec{u}^T\ \frac{\partial \vec{g}_1}{\partial w_{i_{rs}}} \\ \cdots \\ \vec{u}^T\ \frac{\partial \vec{g}_n}{\partial w_{i_{rs}}} \\ \cdots \\ \vec{u}^T\ \frac{\partial \vec{g}_N}{\partial w_{i_{rs}}} \end{bmatrix} - \lambda_i\ (-1)^{r+s} det(W_{i_{rs}})$$

$$
\begin{aligned}
&= \vec{\pi}^{\,T} \times \left[\begin{array}{c}
\vec{u}^T \times \begin{bmatrix} 0 \\ \cdots \\ \frac{\partial g_{1i}}{\partial w_{i_{rs}}} \\ \cdots \\ 0 \end{bmatrix} \\
\cdots \\
\vec{u}^T \times \begin{bmatrix} 0 \\ \cdots \\ \frac{\partial g_{ni}}{\partial w_{i_{rs}}} \\ \cdots \\ 0 \end{bmatrix} \\
\cdots \\
\vec{u}^T \times \begin{bmatrix} 0 \\ \cdots \\ \frac{\partial g_{Ni}}{\partial w_{i_{rs}}} \\ \cdots \\ 0 \end{bmatrix}
\end{array} \right] - \lambda_i \ (-1)^{r+s} det(W_{i_{rs}}) \\
&= \vec{\pi}^{\,T} \times \begin{bmatrix} u_i \ (x_{1ir} - q_{ir})(x_{1is} - q_{is}) \\ \cdots \\ u_i \ (x_{nir} - q_{ir})(x_{1is} - q_{is}) \\ \cdots \\ u_i \ (x_{Nir} - q_{ir})(x_{1is} - q_{is}) \end{bmatrix} - \lambda_i \ (-1)^{r+s} det(W_{i_{rs}}) \\
&= \sum_{n=1}^{N} \pi_n \ (x_{nir} - q_{ir})(x_{1is} - q_{is}) - \lambda_i \ (-1)^{r+s} det(W_{i_{rs}})
\end{aligned}
$$

After setting the above equation to zero and going through procedure similar to that in Section 4.4.1.2, we get

$$W_i^* = (det(C_i))^{\frac{1}{K_i}} \ C_i^{-1} \tag{4.60}$$

where $C_{i_{rs}} = \sum_{n=1}^{N} \pi_n \ (x_{nir} - q_{ir})(x_{1is} - q_{is}), r, s = 1, \cdots, K_i$.

This solution also matches our intuition nicely, since W_i is inversely proportional to the covariance matrix C_i.

4.4.2.3 Optimal Solution for $\vec{u}$

To obtain u_i, set the partial derivative to zero. We then have

$$\frac{\partial L}{\partial u_i} = \sum_{n=1}^{N} \pi_n \ g_{ni} + \lambda \ u_i^{-2} = 0, \ \forall i \tag{4.61}$$

Multiply both sides by u_i and summarize over i. We then have

$$\sum_{i=1}^{I} u_i \ (\sum_{n=1}^{N} \pi_n \ g_{ni}) + \lambda (\sum_{i=1}^{I} \frac{1}{u_i}) = 0 \tag{4.62}$$

Since $\sum_{i=1}^{I} \frac{1}{u_i} = 1$, the optimal λ is

$$\lambda^* = -\sum_{i=1}^{I} u_i \ f_i \tag{4.63}$$

where $f_i = \sum_{n=1}^{N} \pi_n \ g_{ni}$.

After substituting Equation (4.63) into Equation (4.61), we have

$$u_i^2 \ f_i = \sum_{j=1}^{I} u_j \ f_j \tag{4.64}$$

From Equation (4.61), we also know that

$$\lambda = u_1^2 \ f_1 = \cdots = u_i^2 \ f_i = \cdots = u_j^2 \ f_j = \cdots = u_I^2 \ f_I \tag{4.65}$$

That is,

$$u_j = u_i \sqrt{\frac{f_i}{f_j}}, \ \forall j \tag{4.66}$$

After substituting this equation into Equation (4.64), we obtain the optimal solution for u_i:

$$f_i \ u_i^2 - (\sum_{j=1}^{I} \sqrt{f_i \ f_j}) \ u_i = 0 \tag{4.67}$$

$$u_i^* = \sum_{j=1}^{I} \sqrt{\frac{f_j}{f_i}} \tag{4.68}$$

where we discard the solution that $u_i = 0$, because we know $u_i > 0$. Recall that $f_i = \sum_{n=1}^{N} \pi_n \, g_{ni}$, this solution also closely matches our intuition. That is, if the total distance (f_i) of representation i is small (meaning it is close to the ideal query), this representation should receive high weight.

The optimal solutions are summarized in the following equations:

$$\vec{q}_i^{\,T^*} = \frac{\pi^{\,T} \, X_i}{\sum_{n=1}^{N} \pi_i} \tag{4.69}$$

$$W_i^* = (det(C_i))^{\frac{1}{K_i}} \, C_i^{-1} \tag{4.70}$$

$$u_i^* = \sum_{j=1}^{I} \sqrt{\frac{f_j}{f_i}} \tag{4.71}$$

where

$$C_{i_{rs}} = \pi_n \, (x_{nir} - q_{ir})(x_{1is} - q_{is}) \tag{4.72}$$

$$f_i = \sum_{n=1}^{N} \pi_n \, g_{ni} \tag{4.73}$$

$$g_{ni} = (\vec{x}_{ni} - \vec{q}_i)^{\,T} W_i (\vec{x}_{ni} - \vec{q}_i) \tag{4.74}$$

$$i = 1, \cdots, I \tag{4.75}$$

$$r, s = 1, \cdots, K_i \tag{4.76}$$

If we compare this set of equations with Equations (4.46)-(4.48), we can see that, unlike Equations (4.46)-(4.48), solutions in this approach are de-coupled from each other. No iteration is needed, and they can be obtained by a single step, which is very desirable for image retrieval applications.

4.4.2.4 Algorithm descriptions and computation complexity analysis

In this subsection, we will give complete algorithms for both parameter updating and ranked retrieval, and we will examine their computation complexity.

Based on Equations (4.69)-(4.74), the algorithm can be summarized as follows:

1. Input: N training samples $\vec{x}_{ni}$ and their corresponding degree-of-relevance vector $\vec{\pi}$.

2. Output: Optimal solutions as defined in Equations (4.69)-(4.71).

3. Procedure:

   (a) Compute $\vec{q}_i$ as defined in Equation (4.69).

   (b) Compute C_i as defined in Equation (4.72).

   (c) Compute W_i as defined in Equation (4.70).

   (d) Compute g_{ni} as defined in Equation (4.74).

   (e) Compute f_i as defined in Equation (4.73).

   (f) Compute $\vec{u}_i$ as defined in Equation (4.71).

Let a/s and m/d denote the number for addition/subtraction and multiplication/division, respectively. The computation complexity of the above procedure is summarized as follows:

- For $\vec{q}_i,\ \forall i$, $a/s = O(N\ (K_i + 1))$ and $m/d = O(N\ K_i)$.
- For $C_i,\ \forall i$, $a/s = O(2N\ K_i^2)$ and $m/d = O(2N\ K_i^2)$.
- For $W_i,\ \forall i$, $a/s = O(K_i^3 + 2N\ K_i^2)$ and $m/d = O(K_i^3 + 2N\ K_i^2)$
- For $g_{ni},\ \forall n\ , \forall i$, $a/s = O(K_i^2 + 2K_i)$ and $m/d = O(K_i^2 + K_i)$.
- For $f_i,\ \forall i$, $a/s = O(N(K_i^2 + 2K_i))$ and $m/d = O(N(K_i^2 + 2K_i))$.
- For $\vec{u}_i,\ \forall i$, $a/s = O(N(K_i^2 + 2K_i))$ and $m/d = O(N(K_i^2 + 2K_i))$

Therefore, for a single representation i, the total computation complexity is $a/s = O(K_i^3 + 3NK_i^2 + 3NK_i + N)$ and $m/d = O(K_i^3 + 3NK_i^2 + 3NK_i)$. For all the representations, $a/s = \sum_{i=1}^{I}\ O(K_i^3 + 3NK_i^2 + 3NK_i + N)$ and $m/d = \sum_{i=1}^{I}\ O(K_i^3 + 3NK_i^2 + 3NK_i)$. Therefore, the computation complexity is polynomial in all I, N, and K_i. To give readers a more concrete feeling for the complexity, we will plug in some real-world values. Let $I = 5$, $K_i = 10$, and $N = 10$. We will need only 4310 additions/subtractions and 4300 multiplications/divisions to estimate the optimal parameters.

Once we have found the optimal solutions for $\vec{q}_i$, W_i, and $\vec{u}$, we can then perform the ranked retrieval by computing the distance between an image and the query:

$$d_m = \vec{u}^T \vec{g}_m \tag{4.77}$$

$$\vec{g}_m = [g_{m1}, \cdots, g_{mi}, \cdots, g_{mI}] \tag{4.78}$$

$$g_{mi} = (\vec{r}_{mi} - \vec{q}_i)^T W_i (\vec{r}_{mi} - \vec{q}_i) \tag{4.79}$$

$$m = 1, \cdots, M \tag{4.80}$$

$$i = 1, \cdots, I \tag{4.81}$$

where M is the total number of images in the database and $\vec{r}_{mi}$ the i^{th} representation vector for the m^{th} image in the database. The algorithm can then be summarized as follows:

1. Input: Image representation vectors $\vec{r}_{mi}$, query vectors $\vec{q}_i$, and weights W_i and $\vec{u}$.

2. Output: Distance values for all the images in the database.

3. Procedure:

   (a) Compute g_{mi} as defined in Equation (4.79).

   (b) Compute the overall distance d_m according to Equation (4.77).

The computation complexity of the above procedure can be summarized as follows:

- For g_{mi}, $\forall m$, $\forall i$, $a/s = O(K_i^2 + 2K_i)$ and $m/d = O(K_i^2 + K_i)$.

- For d_m, $\forall m$, $a/s = O(K_i^2 + 2K_i + I)$ and $m/d = O(K_i^2 + K_i + I)$.

Therefore, for all the images in the database, we need $a/s = \sum_{m=1}^{M} O(K_i^2 + 2K_i + I)$ and $m/d = \sum_{m=1}^{M} O(K_i^2 + K_i + I)$. Therefore, the computation complexity is polynomial in all I, N, and K_i. Again, to give readers a more concrete feeling for the complexity, we will plug in some real-world values. Let $I = 5$, $K_i = 10$, and $M = 10\ 000$. We will need 1 250 000 additions/subtractions and the same number of multiplications/divisions.

As we can see, a very large portion of computation complexity is not from parameter estimation but from distance computing. This suggests two things. One is that the proposed parameter-updating approach is very efficient. The other is that in order to search quickly, a linear scan of the whole database is not acceptable. We need to develop effective multi-dimensional indexing techniques to improve the query responding time.

4.5 Experimental Results

4.5.1 Data sets

In the experiments reported in this section, the proposed algorithms are tested on one or more of the following data sets.

- MESL data set: This image collection is provided by the Fowler Museum of Cultural History at the University of California-Los Angeles. It contains 286 ancient African and Peruvian artifacts and is part of the Museum Educational Site Licensing Project (MESL), sponsored by the Getty Information Institute.

- Corel data set: The second image collection is obtained from Corel Corporation. It contains more than 70 000 images covering a wide range of more than 500 categories. The 120×80 resolution images are available at *http://corel.digitalriver.com/commerce /photostudio/catalog.htm.*

- Vistex data set A: This data set consists of 384 texture images. The original 24 512×512 texture images are obtained from MIT Media Lab at *ftp://whitechapel.media.mit.edu/pub /VisTex/.* Each image is then cut into 16 128×128 nonoverlap small images. The 16 images from the same big image are considered to be relevant images.

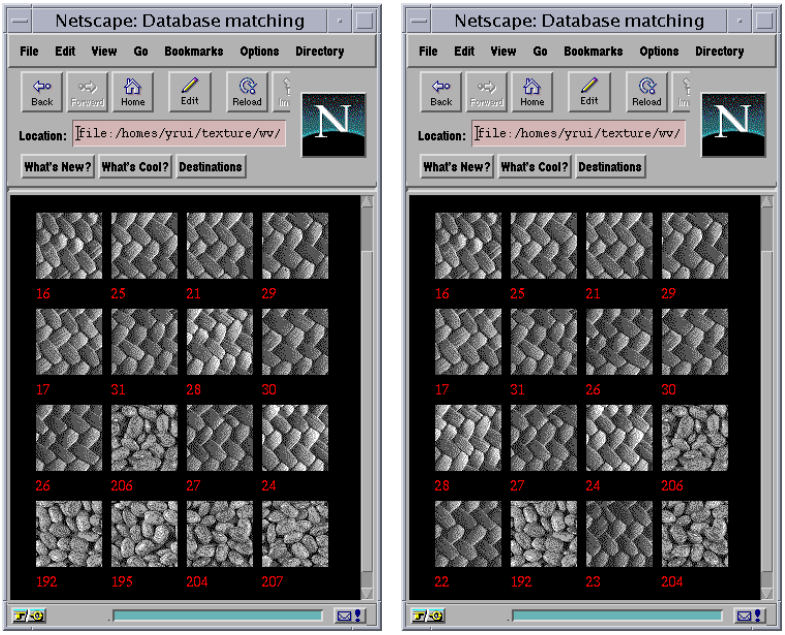
- Vistex data set B: The same as Vistex data set A, but has more images. The 832 small images are obtained from 52 big images.

- MPEG-7 data set: This data set is for MPEG-7 proposal evaluation. We have chosen 300 images to test our algorithms.

4.5.2 Experiments for algorithms in Sections 4.2.1 and 4.2.2

These experiments are to test the heuristic based approaches to updating $\vec{w}_i$ and $\vec{q}_i$. The test data set is Vistex data set A. The representation used are co-occurrence matrix texture representation and wavelet texture representation [138]. Each of the 384 images is selected as the query image, and the 15 best matches are returned. Typical retrieval results are shown in Figure 4.4. The average retrieval precision of 384 query images is summarized in Table 4.1, where the precision is defined as

$$precision = \frac{relevant \;\; images}{returned \;\; images} \times 100\% \tag{4.82}$$



(a) (b)

Figure 4.4 (a) Before relevance feedback (b) After relevance feedback

There are four columns in the table. The term *0 rf* stands for no relevance feedback; *1 rf* corresponds to 1 iteration of relevance feedback; and so forth.

Table 4.1 Retrieval precision (wv: wavelet based; co: co-occurrence matrix based).

	0 rf	1 rf	2 rf	3 rf
wv($\vec{q}i$)	77.27	82.33	85.13	85.53
wv($\vec{w}_i$)	77.67	80.27	80.47	80.53
co($\vec{q}_i$)	57.80	63.47	65.13	66.40
co($\vec{q}_i$)	44.33	48.53	48.80	48.80

The purpose of the experiments is not to compare one texture representation with another, but rather to compare the retrieval performance with relevance feedback versus the performance without relevance feedback. Three observations can be made:

1. The retrieval precision is considerably more improved in the feedback case than in the nonfeedback case.

2. The precision increase in the first iteration of feedback is the largest. Subsequent feedbacks will only achieve minor improvement in precision. This is a very desirable property, because this will guarantee that an acceptable retrieval result is achieved within a limited number of feedback cycles.

3. Updating both the query vectors ($\vec{q}_i$) and the weights ($\vec{w}_i$) can improve the retrieval performance.

4.5.3 Experiments for algorithm in Section 4.2.3

These experiments test the heuristic-based parameter updating algorithm for $\vec{u}$. The data sets are MESL and Corel. We have chosen these two test sets because they complement each other. The size of the MESL test set is relatively small, but it allows us to meaningfully explore all the color, texture, and shape features simultaneously. On the other hand, although the heterogeneity of the Corel test set makes extracting some features (such as shape) difficult, it has the advantages of large size and wide coverage. We believe that testing our proposed approach on both sets will provide a fair evaluation of the performance of our method.

For the MESL test set, the visual features used are color, texture, and shape of the objects in the image. That is,

$$F = \{f_i\} = \{\text{color, texture, shape}\} \tag{4.83}$$

The representations used are color histogram and color moments [28] for the color feature; Tamura [47, 48] and co-occurrence matrix [45, 57] texture representations for the texture feature, and Fourier descriptor and chamfer shape descriptor [29] for the shape feature.

$$\begin{aligned} R = \{\vec{r}_{ij}\} &= \{r_1, r_2, r_3, r_4, r_5, r_6\} \\ &= \{\text{color histogram, color moments, Tamura, co-occurrence matrix,} \\ &\quad \text{Fourier descriptor, chamfer shape descriptor}\} \end{aligned}$$

For the Corel test set, the visual features used are color and texture. That is,

$$F = \{f_i\} = \{\text{color, texture}\} \tag{4.84}$$

The representations used are color histogram and color moments [28] for color feature; and co-occurrence matrix [45, 57] texture representation for texture feature:

$$\begin{aligned} R = \{\vec{r}_{ij}\} &= \{r_1, r_2, r_3\} \\ &= \{\text{color histogram, color moments, co-occurrence matrix}\} \end{aligned}$$

Our proposed relevance feedback architecture is an *open* retrieval architecture. Other visual features or feature representations can be easily incorporated, if needed. The similarity measures used for the corresponding representations are the following. Color Histogram Intersection [28] is used for the color histogram representation; weighted Euclidean is used for the color moments, Tamura texture, co-occurrence matrix, and Fourier shape descriptor [29] representations; and Chamfer matching [29] is used for the chamfer shape representation.

There are two sets of experiments reported here. The first set is on the efficiency of the retrieval algorithm, i.e., how fast the retrieval results converge to the true results (objective test). The second set of experiments is on the effectiveness of the retrieval algorithm, i.e., how good the retrieval results are subjectively (subjective test).

4.5.3.1 Efficiency of the algorithm

The only assumption that we make in the experiments is that the user is consistent when doing relevance feedback. That is, the user does not change his or her information need during the feedback process. With this assumption, the feedback process can be simulated by a computer.

As we have shown in Figure 4.1, the image object is modeled by the combinations of representations with their corresponding weights. If we fix the representations, then a query can be completely characterized by the set of weights embedded in the query object Q. Let set s_1 be the *highly relevant* set, set s_2 the *relevant* set, set s_3 the *no-opinion* set, set s_4 the *nonrelevant* set, and set s_5 the *highly nonrelevant* set. The testing procedure is described as follows:

1. Retrieval results of the ideal case: Let $\vec{u}^*$ be the set of weights associated with the query object Q. The retrieval results based on $\vec{u}^*$ are the ideal case and serve as the baseline for comparing other nonideal cases.

   (a) Specify a set of weights, $\vec{u}^*$, to the query object.

   (b) Set $\vec{u} = \vec{u}^*$.

   (c) Invoke the retrieval algorithm (see Section 4.1).

   (d) Obtain the best N_{RT} returns, RT^*.

   (e) From RT^*, find the sizes of sets $s_i, i = 1, ..., 5$, $n_i, i = 1, ..., 5$. The s_i's are marked by human for testing purpose.

   (f) Calculate the ideal weighted relevant count as

   $$count^* = 3 \times n_1 + 1 \times n_2 \tag{4.85}$$

   Note that 3 and 1 are the scores of the *highly relevant* and *relevant* sets, respectively (see Section 4.2.3). Therefore, $count^*$ is the maximal achievable weighted relevant count and serves as the baseline for comparing other non-ideal case.

2. Retrieval results of relevance feedback case: In the real retrieval situation, neither the user nor the computer knows the specified weights $\vec{u}^*$. However, the proposed retrieval algorithm will move the initial weights $\vec{u}^0$ to the ideal weights $\vec{u}^*$ via relevance feedback.

   (a) Set $\vec{u} = \vec{u}^0$.

   (b) Set the maximum number of iterations of relevance feedback, P_{fd}.

   (c) Initialize the iteration counter, $p_{fd} = 0$.

   (d) Invoke the retrieval algorithm and get back the best N_{RT} returns, $RT(p_{fd})$ (see Section 4.1).

   (e) Compute the weighted relevant count for the current iteration:

$$count(p_{fd}) = 3 \times n_1(p_{fd}) + 1 \times n_2(p_{fd}) \tag{4.86}$$

   where $n_1(p_{fd})$ and $n_2(p_{fd})$ are the number of *highly relevant* and *relevant* objects in $RT(p_{fd})$. These two numbers can be determined by comparing $RT(p_{fd})$ against RT^*.

   (f) Compute the convergence ratio $CR(p_{fd})$ for the current iteration:

$$CR(p_{fd}) = \frac{count(p_{fd})}{count^*} \times 100\% \tag{4.87}$$

   (g) Set $p_{fd} = p_{fd} + 1$. If $p_{fd} \geq P_{fd}$, quit; otherwise continue.

   (h) Feedback the current 5 sets $s_i, i = 1, ..., 5$, to the retrieval system.

   (i) Update the weights $\vec{u}$ according to Equations (4.14)-(4.16). Go to step 2(d).

There are three parameters that affect the behavior of the retrieval algorithm: number of feedbacks P_{fd}, number of returns N_{RT}, and specified query weights $\vec{u}^*$. For P_{fd}, the more relevance feedback iterations, the better the retrieval performance. However, we cannot expect the user to do relevance feedback forever. In the experiments reported here, we set $P_{fd} = 3$ to study the convergence behavior of the first three iterations. The experiments show that the greatest CR increase occurs in the first iteration of feedback, which is a very desirable property.

In all the experiments reported here, for both the MESL and the Corel test sets, 100 randomly selected images are used as the query images, and the values of CR listed in the tables are the averages of the 100 cases.

CR as a function of $\vec{u}^*$

In the MESL test set, there are six representations as described at the beginning of this section. Therefore, both $\vec{u}^*$ and $\vec{u}^0$ have six elements. In addition,

$$\vec{u}^{0^T} = [\frac{1}{6}\ \frac{1}{6}\ \frac{1}{6}\ \frac{1}{6}\ \frac{1}{6}\ \frac{1}{6}\ ] \tag{4.88}$$

where each entry in the vector $\vec{u}^0$ is the weight for its corresponding representation.

In the Corel test set, there are three representations as described at the beginning of this section. Therefore, both $\vec{u}^*$ and $vecu^0$ have three components. In addition,

$$\vec{u}^{0^T} = [\frac{1}{3}\ \frac{1}{3}\ \frac{1}{3}\ ] \tag{4.89}$$

where each entry in the vector $\vec{u}^0$ is the weight for its corresponding representation.

Obviously, the retrieval performance is affected by the offset of the specified weights $\vec{u}^*$ from the initial weights $\vec{u}^0$. We classify $\vec{u}^*$ into two categories (moderate offset and significant offset) by considering how far away they are from the initial weights $\vec{u}^0$.

For the MESL test set, the six moderate offset testing weights are

$$\begin{aligned}
\vec{u}_1^{*^T} &= [0.5\ 0.1\ 0.1\ 0.1\ 0.1\ 0.1] \\
\vec{u}_2^{*^T} &= [0.1\ 0.5\ 0.1\ 0.1\ 0.1\ 0.1] \\
\vec{u}_3^{*^T} &= [0.1\ 0.1\ 0.5\ 0.1\ 0.1\ 0.1] \\
\vec{u}_4^{*^T} &= [0.1\ 0.1\ 0.1\ 0.5\ 0.1\ 0.1] \\
\vec{u}_5^{*^T} &= [0.1\ 0.1\ 0.1\ 0.1\ 0.5\ 0.1] \\
\vec{u}_6^{*^T} &= [0.1\ 0.1\ 0.1\ 0.1\ 0.1\ 0.5]
\end{aligned}$$

The six significant offset testing weights are

$$\vec{u}_7^{*^T} = [0.75\ 0.05\ 0.05\ 0.05\ 0.05\ 0.05]$$

Table 4.2 MESL Moderate Offset Convergence Ratio with $N_{RT} = 12$.

weights	0 feedback	1 feedback	2 feedbacks	3 feedbacks
$\vec{u}_1^*$	88.4	99.7	99.7	99.8
$\vec{u}_2^*$	64.0	98.4	98.5	98.0
$\vec{u}_3^*$	80.7	95.4	93.2	95.8
$\vec{u}_4^*$	56.5	94.9	94.9	94.0
$\vec{u}_5^*$	66.9	87.9	88.1	88.0
$\vec{u}_6^*$	83.5	95.4	97.4	97.5
Avg	73.3	95.3	95.3	95.5

$$\vec{u}_8^{*T} = [0.05\ 0.75\ 0.05\ 0.05\ 0.05\ 0.05]$$

$$\vec{u}_9^{*T} = [0.05\ 0.05\ 0.75\ 0.05\ 0.05\ 0.05]$$

$$\vec{u}_{10}^{*T} = [0.05\ 0.05\ 0.05\ 0.75\ 0.05\ 0.05]$$

$$\vec{u}_{11}^{*T} = [0.05\ 0.05\ 0.05\ 0.05\ 0.75\ 0.05]$$

$$\vec{u}_{12}^{*T} = [0.05\ 0.05\ 0.05\ 0.05\ 0.05\ 0.75]$$

For the Corel test set, the three moderate offset testing weights are

$$\vec{u}_1^{*T} = [0.6\ 0.2\ 0.2]$$

$$\vec{u}_2^{*T} = [0.2\ 0.6\ 0.2]$$

$$\vec{u}_3^{*T} = [0.2\ 0.2\ 0.6]$$

The three significant offset testing weights are

$$\vec{u}_4^{*T} = [0.8\ 0.1\ 0.1]$$

$$\vec{u}_5^{*T} = [0.1\ 0.8\ 0.1]$$

$$\vec{u}_6^{*T} = [0.1\ 0.1\ 0.8]$$

The experimental results for these cases are summarized in Tables 4.2-4.5.

To better represent the process of convergence, we redraw the average CR of the MESL test set and the Corel test set in Figure 4.5.

Based on the tables and figures, some observations can be made:

Table 4.3 MESL Significant Offset Convergence Ratio with $N_{RT} = 12$.

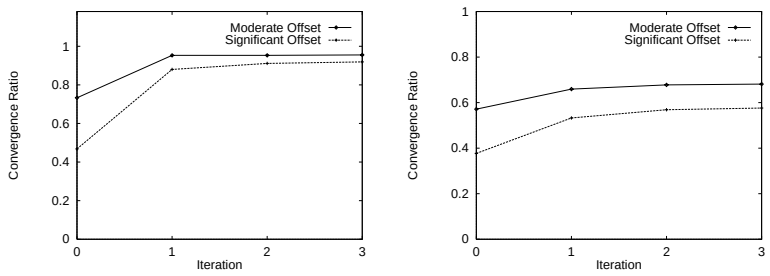
weights	0 feedback	1 feedback	2 feedbacks	3 feedbacks
$\vec{u}_7^*$	40.8	89.6	97.4	98.5
$\vec{u}_8^*$	38.9	95.7	98.8	99.0
$\vec{u}_9^*$	39.0	77.7	74.2	77.3
$\vec{u}_{10}^*$	34.8	91.9	94.6	94.1
$\vec{u}_{11}^*$	62.9	85.7	87.1	87.1
$\vec{u}_{12}^*$	64.1	87.6	94.4	95.3
Avg	46.8	88.0	91.1	91.9

Table 4.4 Corel Moderate Offset Convergence Ratio with $N_{RT} = 1000$.

weights	0 feedback	1 feedback	2 feedbacks	3 feedbacks
$\vec{u}_1^*$	57.1	71.7	74.7	75.2
$\vec{u}_2^*$	55.2	54.7	54.6	54.6
$\vec{u}_3^*$	59.1	71.4	74.0	74.5
Avg	57.1	65.9	67.8	68.1

Table 4.5 Corel Significant Offset Convergence Ratio with $N_{RT} = 1000$.

weights	0 feedback	1 feedback	2 feedbacks	3 feedbacks
$\vec{u}_4^*$	40.2	63.4	68.8	70.0
$\vec{u}_5^*$	31.1	34.4	35.1	35.3
$\vec{u}_6^*$	41.8	62.0	66.7	67.6
Avg	37.7	53.3	56.9	57.6



(a) (b)

Figure 4.5 Convergence ratio curves: (a) MESL test set, (b) Corel test set

- In all the cases, CR increases the most in the first iteration. Later iterations result in only minor increase in CR. This is a very desirable property, which ensures that the user gets reasonable results after only one iteration of feedback. No further feedbacks are needed, if time is a concern.

- CR is affected by the degree of offset. The lower the offset, the higher the final absolute CR. However, the higher the offset, the higher the relative increase of CR.

- Although the finial absolute CR is higher for the MESL test set than for the Corel test set, the final relative increase of CR is comparable for both test sets (around 10%-20%). The convergence process is more challenging for the Corel test set, because of its larger size and fewer number of feature representations.

CR as a function of N_{RT}

N_{RT} is related to the size of the test data set. Normally only 2%-5% of the whole data set is needed. For the MESL test set, we test $N_{RT} = 10, ..., 20$. For the Corel test set, we test $N_{RT} = 850, ..., 1100$ The experimental results are listed in Tables 4.6 and 4.7.

Some observations can be made based on the experiments:

- The first iteration's CR increases the most when N_{RT} is large. This is because the larger the number of returns, the greater the feedback and thus the better the retrieval performance.

Table 4.6 Convergence Ratio for MESL test set with $\vec{u}_7^*$.

N_{NT}	0 feedback	1 feedback	2 feedbacks	3 feedbacks
10	40.4	89.7	97.4	98.6
12	40.9	89.6	97.4	98.5
14	40.8	90.2	97.9	98.4
16	40.2	91.3	98.0	98.2
18	40.4	93.2	97.8	98.0
20	40.5	95.9	97.8	97.9

Table 4.7 Convergence Ratio for Corel test set with $\vec{u}_4^*$.

N_{NT}	0 feedback	1 feedback	2 feedbacks	3 feedbacks
850	40.6	63.4	68.5	69.6
900	40.9	63.8	69.0	70.1
950	41.2	64.2	69.6	70.6
1000	41.4	64.6	70.0	71.1
1050	41.6	65.1	70.6	71.7
1100	41.9	65.5	70.9	72.0

- In the second and third iterations, CR is almost independent of different N_{RT}'s. This is because, after first iteration's feedback, most of the desired objects have been found, and later performance is almost independent of N_{RT}.

4.5.3.2 Effectiveness of the algorithm

Previous subsection's experiments focused on the convergence of the algorithm. This subsection will focus on how good the returns are *subjectively*. The only way to perform subjective tests is to ask the user to evaluate the retrieval system subjectively. Extensive experiments have been carried out. Users from various disciplines (such as computer vision, art, library science), as well as users from industry, were invited to compare the retrieval performance between the proposed *interactive* approach and the *isolated* approach. All users rated the proposed approach much higher than the *computer-centric* approach in terms of capturing their perception subjectivity and information need.

A typical retrieval process on the MESL test set is given in Figures 4.6 and 4.7.

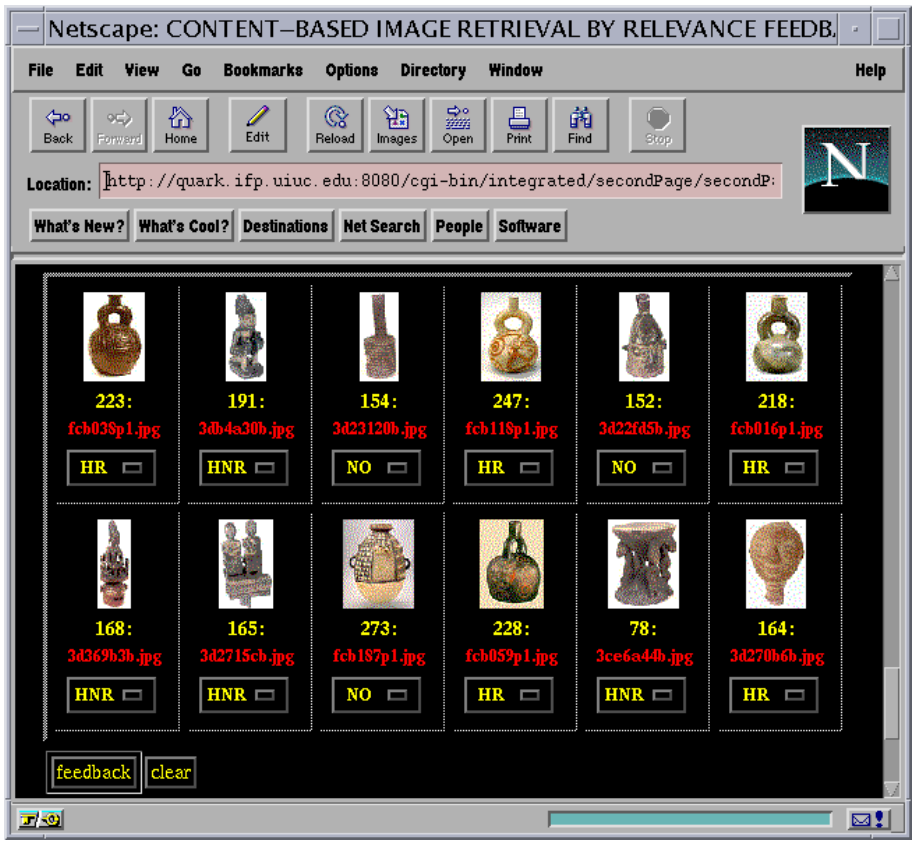


Figure 4.6 The initial retrieval results

The user can browse through the image database. Once he or she finds an image of interest, that image is submitted as a query. As an alternative to this query-by-example mode, the user can also submit images outside the database as queries. In Figure 4.6, the query image is displayed at the upper-left corner and the best 11 retrieved images, with $\vec{u} = \vec{u}^0$, are displayed in order from top to bottom and from left to right. The retrieved results are obtained based on their overall similarities to the query image, which are computed from all the features and all the representations. Some retrieved images are similar to the query image in terms of shape feature while others are similar to the query image in terms of color or texture feature.

Assume the user's true information need is to "retrieve similar images based on their shapes." In the proposed retrieval approach, the user is no longer required to explicitly map an information need to low-level features; rather, the user can express an intended information need by marking the relevance scores of the returned images. In this example, images 247, 218, 228 and

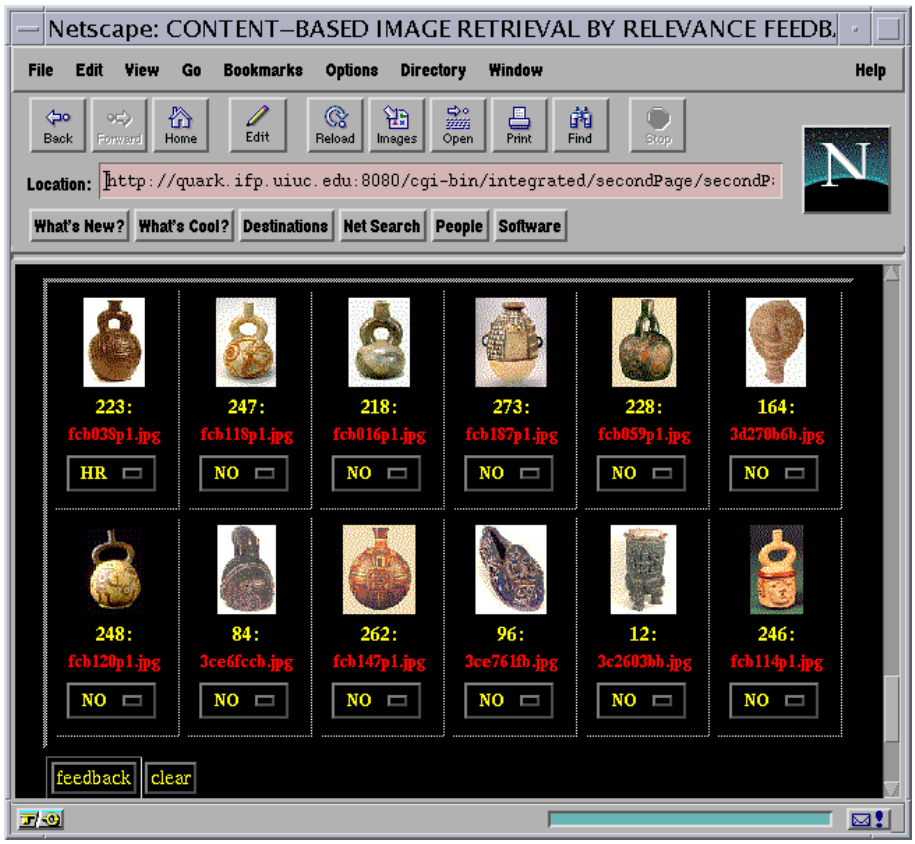


Figure 4.7 The retrieval results after the relevance feedback

164 are marked *highly relevant*. Images 191, 168, 165, and 78 are marked *highly nonrelevant*. Images 154, 152, and 273 are marked *no-opinion*.

Based on the information fed-back by the user, the system *dynamically* adjusts the weights, putting more emphasis on the *shape feature*, possibly even more emphasis to one of the two shape representations which matches user's perception subjectivity of shape. The improved retrieval results are displayed in Figure 4.7. Note that our shape representations are invariant to translation, rotation, and scaling. Therefore, images 164 and 96 are relevant to the query image.

Unlike the *isolated* approach, where the user has to precisely decompose an information need into different features and representations and precisely specify all the weights associated with them, the proposed *interactive* approach allows the user to submit a coarse initial query and

continuously refine the information need via relevance feedback. This approach greatly reduces the user's effort of composing a query and captures the user's information need more precisely.

4.5.4 Experiments for algorithms in Section 4.4.2

We have constructed a CBIR system based on the optimization algorithm developed in Section 4.4.2. Figure 4.8 is an interface of it.

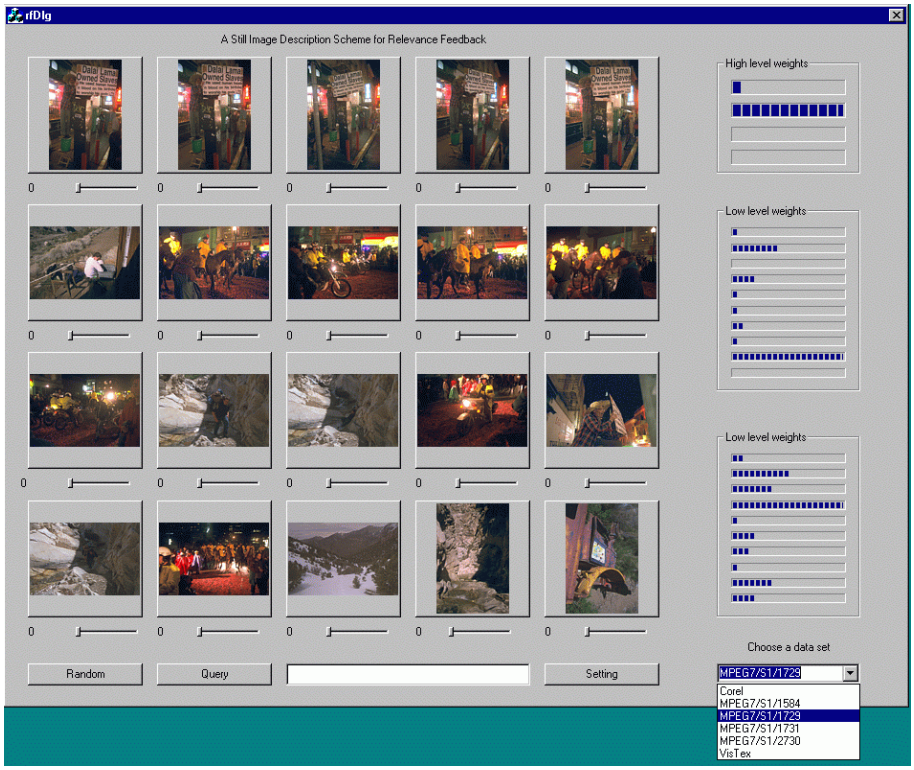


Figure 4.8 The interface of the demo system

A user has many options to configure the system and to choose a particular data set. The system can be configured based on the following settings (Figure 4.9):

- Move query vector versus do not move query vector
- Use full covariance matrix versus use diagonal matrix
- Use any combination of the feature representations

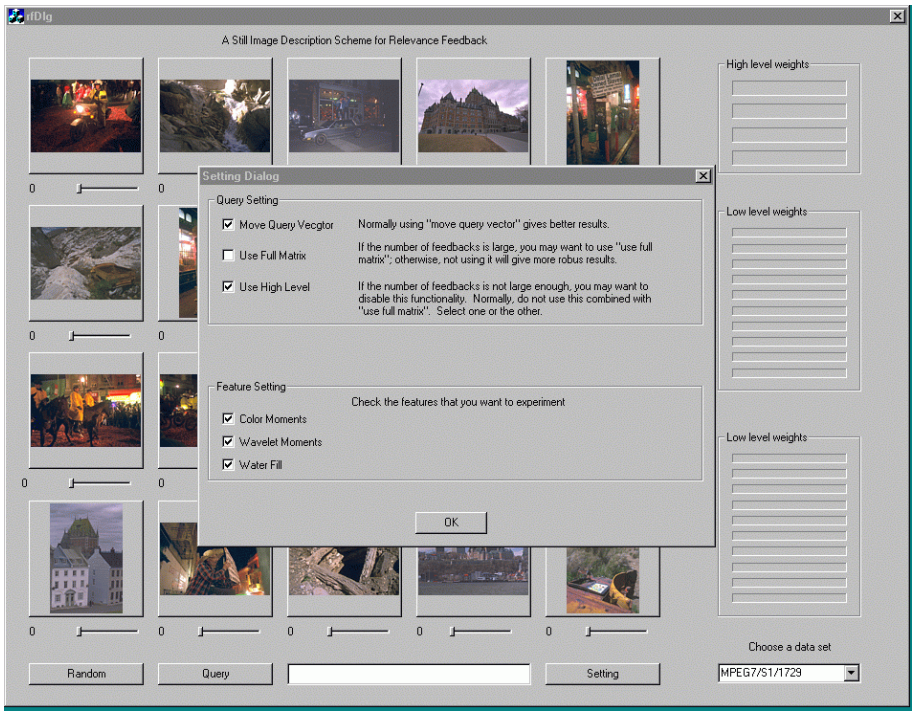


Figure 4.9 The configuration of the demo system

Clicking on the "data set" choice box will give a user the choice of selecting different data sets. (Currently, Vistex data set B, MPEG-7 data set, and Corel data set have been populated into the database.) The progress controls on the right-hand side of the interface display the weights ($\vec{w}_i$ and $\vec{u}$) for a particular query. A typical retrieval result before and after (once and twice) relevance feedback are illustrated in Figures 4.10 to 4.12.

4.6 Conclusions

CBIR has emerged as one of the most active research areas in the past few years. Most of the early research effort focused on finding the "best" image features or representations. Retrieval was performed by summing the distances of individual feature representation with fixed weights. Although this *isolated* approach establishes the basis of CBIR, the usefulness of such systems was limited.

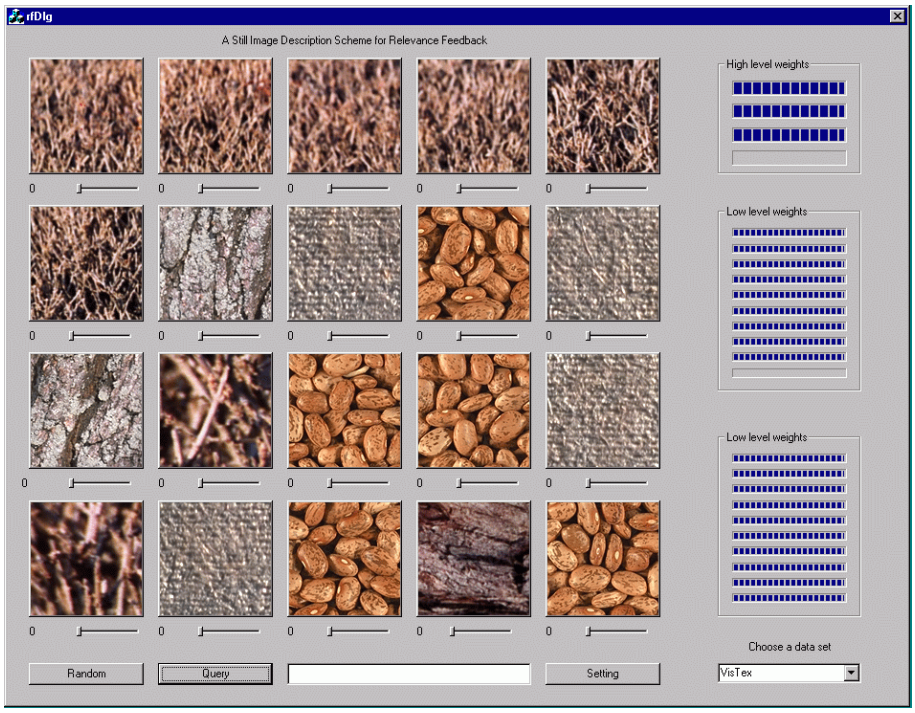


Figure 4.10 Before the relevance feedback

In this chapter, we have introduced a human-computer *interactive* approach to CBIR based on relevance feedback. Unlike the *isolated* approach, where the user has to precisely decompose the information need into precise weights, the *interactive* approach allows the user to submit a coarse initial query and continuously refine his information need via relevance feedback. This approach greatly reduces the user's effort of composing a query and captures the user's information need more precisely.

Based on the description of the relevance feedback algorithm in Sections 4.1-4.4, we can observe its following properties.

- *Multimodality*: The proposed image object model, and therefore the retrieval model, supports multiple features and multiple representations. In contrast to the *isolated* approach's attempt to find the single "best" universal feature representation, the proposed approach concentrates on how to organize the multiple feature representations, such that appropriate feature representations are invoked (emphasized) at the right place and time. The

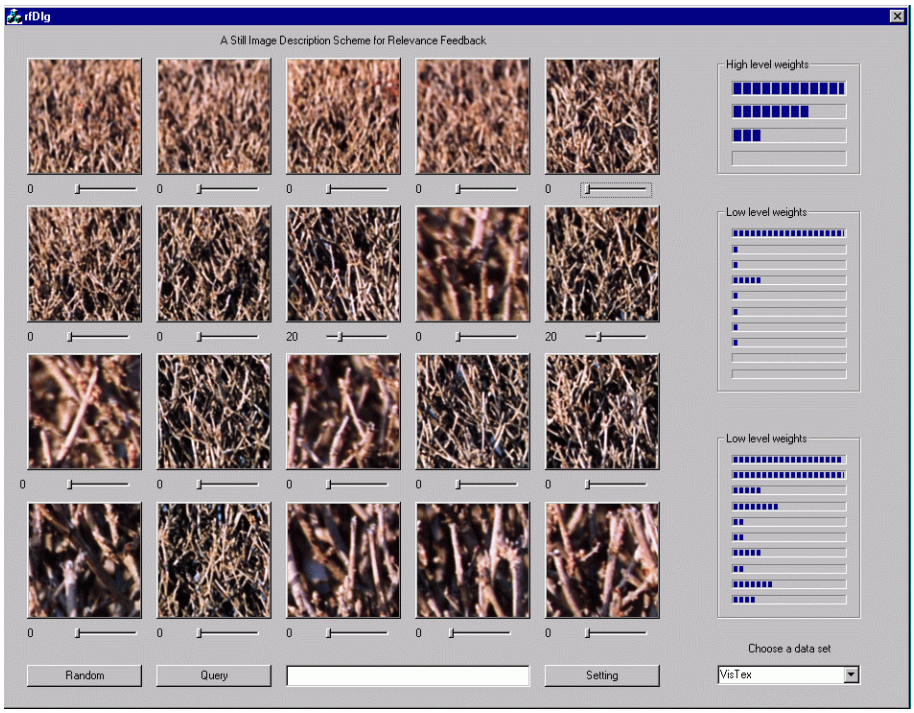


Figure 4.11 After one iteration of relevance feedback

multimodality approach allows the system to better model the user's perception subjectivity.

- *Interactivity*: The proposed approach is *interactive*. The interactivity allows the system to make use of the ability from both the computer and the human.

- *Dynamic*: In contrast to a *isolated* approach's fixed query weights, the proposed approach *dynamically* updates the query weights via relevance feedback. The advantages are for both the user and the system designer. They are no longer required to specify a precise set of weights. Instead, the user interacts with the system, indicating which returns he or she thinks are relevant. Based on the user's feedback, query weights are dynamically updated.

In this chapter, we have explored two general approaches to parameter updating via relevance feedback. One is heuristic-based while the other is optimization based. Based on subjective test, we observe that the latter out-perform the former in retrieval performance.

Figure 4.12 After two iterations of relevance feedback

One important aspect, i.e., normalization, that we did not cover in this chapter is presented in Appendix A.

CHAPTER 5

FUTURE RESEARCH DIRECTIONS

Based on the previous chapters' discussion, we summarize in this chapter some possible future research directions.

5.1 Human in the Loop

A fundamental difference between a computer vision pattern recognition system and an image retrieval system is that a human is an indispensable part in the latter system. We need to explore the synergy of human and computer [1, 2]. This research trend has already been reflected in the evolution of content-based image retrieval. Early literatures emphasize "fully automated system" and try to find a "single best feature." But such an approach does not lead to a success, as the computer vision technique is not there yet. More recent research emphasis is given to "interactive systems" and "human in the loop." The QBIC team uses interactive region segmentation [91]. The MIT team moves from the "automated" Photobook to "interactive" FourEyes [113, 114]. The WebSEEk system allows for dynamic feature vector recomputation based on user's feedback [153]. The UCSB team incorporates supervised learning in texture analysis [134, 135]. The MARS team formally proposes a *relevance feedback* architecture in image retrieval, where human and computer can interact to improve the retrieval performance [138, 29, 24, 23, 159].

5.2 High-Level Concepts and Low-Level Visual Features

Humans tend to use high-level concepts in everyday life. However, what current computer vision techniques can automatically extract from images are mostly low-level features. In a general setting, the low-level features do not have a close link to the high-level concepts. To narrow down this semantic gap, some off-line and on-line processing is needed. The off-line processing can be achieved by using either supervised learning, unsupervised learning, or the combination of the two. Neural Nets, genetic algorithm, and clustering are such learning tools [134, 135, 113, 114]. For on-line processing, a powerful and user-friendly intelligent query interface is needed to perform this task. It should allow the user to easily provide his or her evaluation of current retrieval result to the computer. The relevance feedback technique proposed in MARS is one possible tool [138, 23].

5.3 Web Oriented

The expansion of the World Wide Web is astonishing. Each day thousands of documents, often including images, are added to the Web. To better organize and retrieve the almost unlimited information, Web-based search engines are highly desired. Such solution exists for text-based information. The fact that Yahoo, Inforseek, etc. are among the most frequently visited web sites indicates the need for Web-based search engines [11]. For images on the Web, even though some good work has been taken place [8, 142, 143, 144, 160], technical breakthroughs are needed to make image search engines comparable to their text-based counterparts.

One major technical barrier lies in linking the low-level visual feature indexes used in most systems today to more desirable semantic-level meanings. Based on preliminary on-line experiments, we have observed that subject browsing and text-based matching are still more popular operations than feature-based search options [161]. That is partly why commercial image retrieval systems on the Web typically use customized subject categories to organize their image collection. Usually, different image retrieval systems focus on different sections of users and

content. As a result, the indexing features and the subject taxonomies are also different, causing the concern of interoperability. Several recent efforts in standards have started to address this issue [11, 162]. Several research systems on image metaservers [163, 164, 165] have also investigated frameworks for integrated access to distributed image libraries.

5.4 High-Dimensional Indexing

A by-product of the web expansion is the huge collection of images. Most currently existing research prototype systems only handle hundreds or at most a few thousand of images: therefore, a sequential scan of all the images will not seriously degrade the system's performance. For this reason, only a few existing systems explored the multidimensional indexing aspect of image retrieval [18, 3, 7, 8]. However, because image collections increase in size, retrieval speed is becoming a bottleneck. Although some progress has been made in this area, effective high-dimensional indexing techniques are still in urgent need of exploration.

5.5 Performance Evaluation Criterion

Any technique is pushed forward by its domain's evaluation criterion. Signal to noise ratio (SNR) is used in data compression, and precision and recall are used in text-based information retrieval. Good measure will lead the technique in the correct direction, whereas bad measures will mislead the research effort. Currently, some systems measure performance based on "cost/time" to find the correct images [122]. Other systems evaluate performance using precision and recall, terms borrowed from text-based retrieval. However, images have their own characteristics, such as the perception subjectivity. There is an urgent need to define meaningful image retrieval performance measures, which can take into account the human perception characteristics and can lead the research effort in the correct direction [1, 2].

5.6 Integration of Disciplines and Media

Both the Database community literature and the computer vision community literature have used "image database" as the title of many articles [148]. However, in reality, most database community systems are nonimage (text-based keywords or graphics-based icons) databases, whereas most computer vision systems are image nondatabases (just a large file containing thousands of images is not a database, because most fundamental database units such as data model, indexing, etc. are not addressed at all). To our knowledge, even though there are continuing research efforts to build true image databases [49, 157], the systems are not complete.

A successful image database system requires an interdisciplinary research effort. Besides the integration of database management and computer vision, the research from the traditional information retrieval area [151, 26, 154, 166, 25, 167] is also indispensable. Although the traditional Information Retrieval area's research focus was in text-based document retrieval, many useful retrieval models and techniques can be adapted to image retrieval. Some successful examples of such research effort include the adaption of Boolean retrieval model in image retrieval [137, 49], and the utilization of relevance feedback in image retrieval [138, 29, 24, 23].

Another observation is that integration of multimedia, multimodalities provides great potential for improved indexing and classification of images in general domains. Research in [168, 169, 161] has shown promising results in using both textual and visual features in automatic indexing of images. More sophisticated techniques for cross-mapping image classification between the high level using textual cues and the low level using the visual cues will bear fruits.

PART II

PART II VIDEO SYSTEM

CHAPTER 6

INTRODUCTION TO VIDEO DATABASE SYSTEMS

Research on how to efficiently access the video content has become increasingly active in the past few years [170, 171, 172, 173]. Considerable progress has been made in video analysis, representation, browsing, and retrieval, the four fundamental bases for accessing video content. *Video analysis* deals with the *signal processing* part of the video system, including shot boundary detection, key frame extraction, etc. *Video representation* concerns with the *structure* of the video. An example of the video representations is the tree structured key frame hierarchy [174, 172]. Build on top of the video representation, *video browsing* deals with how to use the representation structure to help the viewers browse the video content. Finally, *video retrieval* concerns about retrieving interesting video objects to the viewer. The relationship between these four research areas is illustrated in Figure 6.1.

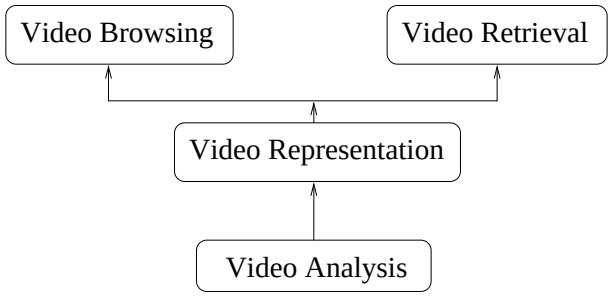


Figure 6.1 Relations between the four research areas

So far, most of the research effort has gone into video analysis. Though it is the basis for all the other research activities, it is not the ultimate goal. Relatively less research exists on

video representation, browsing, and retrieval. From Figure 6.1, video browsing and retrieval are on the very top of the diagram. They *directly* support users' access to the video content. To access a temporal medium, such as a video clip, browsing and retrieval are equally important. Browsing helps a user to quickly grasp the global picture of the whole data, whereas retrieval helps a user to find a specific query's results.

An analogy explains this argument. How does a reader efficiently access a 1000-page book's content? Without reading the whole book, the reader can first go to the book's Table of Contents (ToC), finding which chapters or sections suit his or her need. If he has specific questions (queries) in mind, such as finding a terminology or a key word, he can go to the Index and find the corresponding book sessions containing that question. In short, a book's ToC helps a reader *browse* and a book's index helps a reader *retrieve*. Both aspects are equally important in helping users access the book's content. For current videos, unfortunately, we lack both the ToC and the Index. Techniques are urgently needed for constructing a ToC and Index to facilitate the video access.

We can do an even better job in video than in the book case by integrating video's ToC and Index into a unified framework. For a continuous and long medium such as video, a "back and forth" mechanism between browsing and retrieval is crucial. The video library users may need to browse the video before they know what to retrieve. On the other hand, after retrieving some video objects, the users will be guided to browse the video in the correct direction.

6.1 Terminologies

Before we go into the details of the discussion, it is beneficial to first introduce some important terminologies used in the video research area:

- *Video shot*: An unbroken sequence of frames recorded from a single camera. It is the building block of video streams.

- *Key frame*: The frame that can represent the salient content of a shot. Depending on the content complexity of the shot, one or more key frames can be extracted.

- *Video scene*: A collection of semantically related and temporally adjacent shots, depicting and conveying a high-level concept or story. While shots are marked by physical boundaries, scenes are marked by semantic boundaries.[1]

- *Video group*: An intermediate entity between the physical shots and semantic scenes that serves as the bridge between the two. Examples of groups are temporally adjacent shots [174] or visually similar shots [172].

In summary, the video stream can be structured into a hierarchy consisting five levels: video, scene, group, shot, and key frame, from top to bottom increasing in granularity [173] (see Figure 6.2).

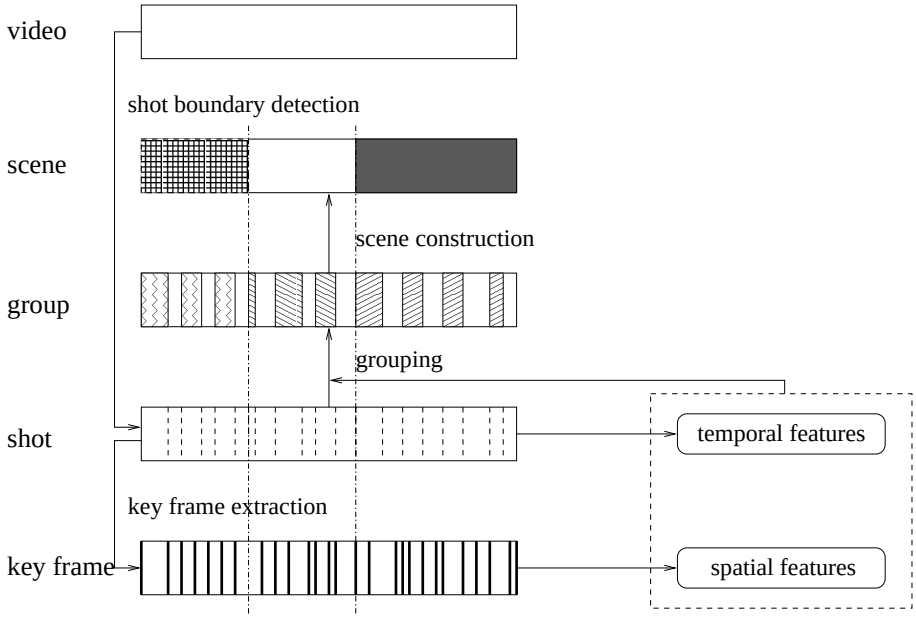


Figure 6.2 A hierarchical video representation

The goal of this and the following chapters is to analyze the strength and weakness of existing techniques, and to explore novel techniques for constructing both the video ToC and

[1]Some of the early literatures in video parsing misused the phrase *scene change detection* for *shot boundary detection*. To avoid any later confusion, we will use *shot boundary detection* for the detection of physical shot boundaries while using *scene boundary detection* for the detection of semantic scene boundaries.

video Index, and to integrate them into a unified framework. In Chapter 7, a brief but necessary review of video analysis and representation is presented. In Chapter 8, a detailed algorithm on semantic-level ToC construction is proposed and described. In Chapter 9, we analyze the various requirements of video retrieval and propose our approach on video Index construction. In Chapter 10, we propose a unified framework to better support video browsing and retrieval. Conclusions and future research directions are summarized in Chapter 11.

CHAPTER 7

VIDEO ANALYSIS AND VIDEO REPRESENTATION

7.1 Video Analysis

As in Figure 6.1, video analysis is the bases for later video processing. It contains *shot boundary detection* and *key frame extraction*.

7.1.1 Shot boundary detection

It is not efficient to process the video clip as a whole. It is beneficial to first decompose the video clip into shots and to perform signal processing at the shot level. In general, automatic shot boundary detection techniques can be classified into five categories: pixel based, statistics based, transform based, feature based, and histogram based. Pixel-based approaches use the pixel-wise intensity difference as the indicator for shot boundaries [170, 175]. One of its drawbacks is its sensitivity to noise. To overcome this problem, Kasturi and Jain propose using intensity statistics (mean and standard deviation) as the shot boundary detection measure [176]. Exploring how to achieve faster speed, Arman et al. [177] propose using the discrete cosine transform (DCT) coefficients in the compressed domain as the boundary measure. Other transform-based shot boundary detection approaches make use of the motion vectors, which are already embedded in the MPEG stream [178, 179]. Zabih et al. [180] address the problem from another angle. The edge features are first extracted from each frame. Shot boundaries are then detected by comparing the edge difference [180]. So far, the histogram-based approach is the

most popular. Several researchers claim that this approach achieves a good tradeoff between accuracy and speed [170]. Representatives of this approach are [181, 182, 170, 183, 184]. A more recent work is based on clustering and postfiltering [185], which achieves fairly high accuracy without many false positives. Two comprehensive comparisons of various shot boundary detection techniques are in [186, 187].

7.1.2 Key frame extraction

After the shot boundaries are detected, corresponding key frames can then be extracted. Simple approaches may just extract the first and last frames of each shot as the key frames [184]. More sophisticated key frame extraction techniques are based on visual content complexity indicator [188], shot activity indicator [189], and shot motion indicator [190].

7.2 Video Representation

Considering that each video frame is a 2D object and the temporal axis makes up the third dimension, a video stream spans a 3D space. Video representation is the *mapping* from the 3D space to the 2D view screen. Different mapping functions characterize different video representation techniques.

7.2.1 Sequential key frame representation

After obtaining shots and key frames, an obvious and simple video representation is to sequentially lay out the key frames of the video, from top to bottom and from left to right. This simple technique works well when the number of key frames is small. When the video clip is long, this technique does not scale, because it does not capture the embedded information within the video clip, except for time.

7.2.2 Group-based representation

To obtain a more meaningful video representation when the video is long, related shots are merged into groups [174, 172]. Zhang et al. [174] divide the whole video stream into multiple video segments, each containing equal numbers of consecutive shots. Each segment is further divided into subsegments; thus constructing a tree-structured video representation. Zhong et al. [172] proposed a cluster-based video hierarchy in which the shots are clustered based on their visual content. This method again constructs a tree-structured video representation.

7.2.3 Scene-based representation

To provide the user with better access to the video, constructing a video representation at a semantic level is needed [173, 171]. In [171], a scene transition graph (STG) of video representation is proposed and constructed. Video sequence is first segmented into shots. Shots are then clustered by using *time-constrained clustering*. STG is then constructed based on the time flow of clusters.

7.2.4 Video mosaic representation

Instead of representing the video structure based on the video-scene-group-shot-frame hierarchy as discussed above, this approach takes a different perspective [191]. The mixed information within a shot is decomposed into three components:

- *Extended spatial information*: This captures the appearance of the entire scene imaged in the video shot and is represented in the form of a few (often just one) panoramic mosaic images constructed by composing the information from the different views of the video shot in the individual frames into a single image.

- *Extended temporal information*: This captures the motion of independently moving objects in the video shot, e.g., in the form of their trajectories.

- *Geometric information*: This captures the 3D video shot structure, as well as the geometric transformations that map the frames to the common mosaic image, which are induced by the motion of the camera.

Taken together, these three components provide a complete and compact description of the video data.

CHAPTER 8

VIDEO BROWSING

Because of video's length and unstructured format, browsing a video efficiently is not an easy task. From the perspective of browsing, video is analogous to a book. Access to a book is greatly facilitated by a well-designed ToC that captures the semantic structure of the book. For current video, a lack of such a ToC makes the task of browsing and retrieval inefficient, where a user searching for a particular object of interest has to use the time-consuming "fast forward" and "rewind" operations. Efficient techniques need to be developed to construct video ToC to facilitate user's access.

Over the past few years, progress has been made in constructing video ToC, but mainly limited to the *shot*, *key frame*, and *group* levels. Because the video ToC at these levels is not closely related to the semantics of the video and normally has a large number of entries, it is still difficult to use. In the case of the shot- and key-frame-based video ToC, for example, it is not uncommon for a modern movie to contain thousands of shots and key frames. This is evidenced in [192] – there are 300 shots in a 15-minute video segment of the movie "Terminator 2 - the Judgment Day," and the movie lasts 139 minutes. Because of the large number of key frames, a simple 1D array presentation of key frames for the underlying video is almost meaningless. More importantly, people watch the video by its semantic scenes, not the physical shots or key frames. Shots cannot convey meaningful semantics unless they are purposely organized into scenes. The video ToC construction at the scene level is thus fundamentally important to video browsing and retrieval.

In this chapter we will present a novel framework in scene-based video ToC construction. Specifically, in Section 8.1, we evaluate some related work in video ToC construction at various levels. In Section 8.2, a novel framework for scene structuring is presented, where shots are organized into groups and groups into semantically meaningful scenes. To put the proposed algorithm into practice, in Section 8.3, the techniques based on Gaussian normalization are proposed to determine the algorithm's parameters. In Section 8.4, the effectiveness of the proposed approach is validated by experiments over real-world movie video clips. Concluding remarks are given in Section 8.5.

8.1 Related Work

Work on extracting video ToC has been done at various levels (key frame, shot, group, and scene). Below, we briefly review and evaluate some of the common approaches proposed.

8.1.1 Shot- and key-frame-based video ToC construction

In this approach, the raw video stream is first segmented into a sequence of shots by using automatic shot boundary detection techniques. Key frames are then extracted from each shot. The video ToC is constructed as a sequence of the key frames. A user can access the video by browsing through the sequence of key frames.

After the shot boundaries are detected and key frames extracted, the sequence of key frames, together with their frame identifiers (ids), are used as the video ToC. Although this approach provides the user with better access to the video than does the unstructured video, it still does not convey enough semantic meaning, as discussed in Chapter 6.

8.1.2 Group-based video ToC construction

To obtain a video ToC at a higher level, related shots are merged into groups, based on which a browsing hierarchy can be constructed [174, 172]. Zhang et al. [174] divide the video stream into multiple video segments, each containing an equal number of consecutive shots. Each

segment is further divided into subsegments; thus constructing a hierarchy of video content that is used to assist browsing. In this approach, time is the only factor considered, and no visual content is used in constructing the browsing hierarchy. In contrast, Zhong et al. [172] proposed a cluster-based video hierarchy, in which the shots are clustered based on their visual content. Although this approach takes the visual content into account, the time factor is lost. One common drawback of the above approaches is that the video structuring is not at a high enough semantic level. Although these group-based video ToC provide better solutions than shot- and key-frame-based video ToC, they still cannot convey the underlying semantic concepts and stories of the video.

8.1.3 Scene-based video ToC construction

To provide the user with better access to the video, construction of video ToC at a semantic level is needed. Approaches to scene-based video ToC construction can be classified into two categories, *model based* and *general purpose*. In the model-based approach, an *a priori* model of a particular application or domain is first constructed. Such a model specifies the scene boundary characteristics, based on which unstructured video stream can be abstracted into a structured representation. The theoretical framework of this approach was proposed by Swangberg et al. [182], and it has been successfully realized in many interesting applications, including news video parsing [193] and TV soccer program parsing [194]. Because the video parsing is based on the domain model, this approach normally achieves high accuracy. One of the drawbacks of this approach, however, is that for each application a domain model needs to be constructed before the parsing process can proceed. The modeling process is time consuming and requires good domain knowledge and experience.

Another approach to scene-based video ToC construction does not require an explicit domain model. Two of the pioneering works of this approach are from Princeton University [179, 195, 192, 171] and Toshiba Corp. [196]. In [192, 171], the video stream is first segmented into shots. Then *time-constrained clustering* is used to construct visually similar and temporally adjacent

shot clusters. Finally a scene transition graph (STG) is constructed based on the clusters and *cutting edges* are identified to construct the scene structure. In [196], instead of using a STG, the authors group shots of alternating patterns into scenes (they call *acts*). A 2D presentation of the video structure is then created, with scenes displayed vertically, and key frames displayed horizontally. This video ToC provides the user a much more meaningful way of accessing the video content. The advantages of scene-based video ToC over the other approaches are

- The other approaches produce too many entries to be efficiently presented to the viewer.
- Shots, key frames, and even groups convey only physical discontinuity, whereas scenes convey semantic discontinuity, such as scene change in time and/or location.

8.2 The Proposed Approach to Scene-Based Video ToC Construction

The proposed approach to video ToC construction consists of four modules: shot boundary detection and key frame extraction, spatio-temporal feature extraction, time-adaptive grouping, and scene structure construction. We will discuss each of the modules in turn.

8.2.1 Shot boundary detection and key frame extraction

As described in Chapter 7, a lot of work has been done in shot boundary detection, and many of the approaches achieve satisfactory performance [197]. In this paper, we use an approach similar to the one used in [170]. For key frame extraction, although more sophisticated techniques exist [189, 190], they require high computation effort. We select the beginning and ending frames of a shot as the two key frames to achieve fast processing speed.

8.2.2 Spatio-temporal feature extraction

At the shot level, the shot activity measure is extracted to characterize the temporal information:

$$Act_i = \frac{1}{N_i - 1} \sum_{k=1}^{N_i - 1} Diff_{k,k-1} \tag{8.1}$$

$$Diff_{k,k-1} = Dist(Hist(k), Hist(k-1)) \tag{8.2}$$

where Act_i and N_i are the activity measure and number of frames for shot i; $Diff_{k,k-1}$ is the color histogram difference between frames k and $k-1$; $Hist(k)$ and $Hist(k-1)$ are the color histograms for frames k and $k-1$; $Dist()$ is a distance measure between histograms. We adopt the intersection distance [28] in this paper. The color histograms used are 2D histograms along the H and S axes in HSV color space. We disregard V component because of its sensitivity to the lighting condition.

At the key frame level, visual features are extracted to characterize the spatial information. In the current algorithm, color histograms of the beginning and ending frames are used as the visual feature for the shot:

$$Hist(b_i) \tag{8.3}$$

$$Hist(e_i) \tag{8.4}$$

where b_i and e_i are the beginning and ending frames of shot i.

Based on the above discussion, a shot is modeled as

$$shot_i = shot_i(b_i, e_i, Act_i, Hist(b_i), Hist(e_i)) \tag{8.5}$$

which captures both the spatial and the temporal information of a shot. At higher levels, this spatial-temporal information is used in grouping and scene structure construction.

8.2.3 Time-adaptive grouping

Before we construct the scene structure, it is convenient to first create an intermediate entity *group* to facilitate the later process. In grouping, similar shots are organized into groups, because similar shots have a high possibility of being in the same scene. For shots to be similar, the following properties should be satisfied:

- Visual similarity: Similar shots should be visually similar. That is, they should have similar spatial ($Hist(b_i)$ and $Hist(e_i)$) and temporal (Act_i) features.

- Time locality: Similar shots should be close to each other temporally [192]. For example, visually similar shots, if far apart from each other in time, seldom belong to the same scene and, hence, not to the same group.

Yeung et al. [192] proposed a *time-constrained clustering* approach to grouping shots, where the similarity between two shots is set to 0 if their time difference is greater than a predefined threshold. We propose a more general *time-adaptive grouping* approach based on the two properties for similar shots described above. In our proposed approach, the similarity of two shots is an increasing function of visual similarity and a decreasing function of frame difference. Let i and j be the indexes for the two shots whose similarity is to be determined, where *shot* j > *shot* i. The calculation of the shot similarity is described as follows:

1. Calculate the shot color similarity $ShotColorSim$:

   (a) Calculate the four raw frame color similarities: $FrameColorSim_{b_j,e_i}$, $FrameColorSim_{e_j,e_i}$, $FrameColorSim_{b_j,b_i}$, and $FrameColorSim_{e_j,b_i}$, where $FrameColorSim_{x,y}$ is defined as

   $$FrameColorSim_{x,y} = 1 - Diff_{x,y} \tag{8.6}$$

   where x and y are two arbitrary frames.

   (b) To model the importance of time locality, we introduce the concept of *temporal attraction*, $Attr$, which is a decreasing function of the frame difference:

   $$Attr_{b_j,e_i} = max(0, 1 - \frac{b_j - e_i}{baseLength}) \tag{8.7}$$

   $$Attr_{e_j,e_i} = max(0, 1 - \frac{e_j - e_i}{baseLength}) \tag{8.8}$$

   $$Attr_{b_j,b_i} = max(0, 1 - \frac{b_j - b_i}{baseLength}) \tag{8.9}$$

   $$Attr_{e_j,b_i} = max(0, 1 - \frac{e_j - b_i}{baseLength}) \tag{8.10}$$

   $$baseLength = MULTIPLE * avgShotLength \tag{8.11}$$

   where $avgShotLength$ is the average shot length of the whole video stream; $MULTIPLE$ is a constant that controls how fast the temporal attraction will decrease to 0. For

our experiment data, we find $MULTIPLE = 10$ gives good results. The above definition of *temporal attraction* says that the farther apart the frames, the less the *temporal attraction*. If the frame difference is larger than $MULTIPLE$ times the average shot length, the attraction decreases to 0.

(c) Convert the raw similarities to *time-adaptive* similarities, which capture both the visual similarity and time locality:

$$FrameColorSim'_{b_j,e_i} = Attr_{b_j,e_i} \times FrameColorSim_{b_j,e_i} \quad (8.12)$$

$$FrameColorSim'_{e_j,e_i} = Attr_{e_j,e_i} \times FrameColorSim_{e_j,e_i} \quad (8.13)$$

$$FrameColorSim'_{b_j,b_i} = Attr_{b_j,b_i} \times FrameColorSim_{b_j,b_i} \quad (8.14)$$

$$FrameColorSim'_{e_j,b_i} = Attr_{e_j,b_i} \times FrameColorSim_{e_j,b_i} \quad (8.15)$$

(d) The color similarity between shots i and j is defined as the maximum of the four frame similarities:

$$\begin{aligned} ShotColorSim_{i,j} = max(\quad & FrameColorSim'_{b_j,e_i}, \\ & FrameColorSim'_{e_j,e_i}, \\ & FrameColorSim'_{b_j,b_i}, \\ & FrameColorSim'_{e_j,b_i} \quad) \end{aligned}$$

2. Calculate the shot activity similarity $ShotActSim$:

$$ShotActSim_{i,j} = Attr_{center} \times |Act_i - Act_j| \quad (8.16)$$

$$Attr_{center} = max(0, 1 - \frac{(b_j + e_j)/2 - (b_i + e_i)/2}{baseLength}) \quad (8.17)$$

where $Attr_{center}$ is the *temporal attraction* between the two center frames of shot i and shot j.

3. Calculate the overall shot similarity $ShotSim$:

$$ShotSim_{i,j} = W_C * ShotColorSim_{i,j} + W_A * ShotActSim_{i,j} \quad (8.18)$$

where W_C and W_A are appropriate weights for color and activity measures.

8.2.4 Scene structure construction

Similar shots are organized into a group, but even dissimilar groups can be grouped into a single scene if they are semantically related. Video is a sequential medium. Therefore, even though two or more processes develop simultaneously in a video, they have to be displayed sequentially, one after another. This is common in movies. For example, when two people talk to each other, even though both people contribute to the conversation, the movie switches back and forth between these two people. In this example, there clearly exist two groups, one corresponding to person A, and the other corresponding to person B. Even though these two groups are dissimilar groups, they are semantically related and should be merged together into a single scene. The proposed scene structure construction algorithm achieves this goal using a two-step process:

- collect similar shots into groups using *time-adaptive grouping*, and
- merge semantically related groups into a unified scene.

The detailed algorithm is described as follows:

Main procedure

- Input: Video shot sequence, $S = \{shot\ 0, ..., shot\ i\}$.
- Output: Video structure in terms of *scene*, *group*, and *shot*.
- Procedure:
  1. Initialization: Assign shot 0 to group 0 and scene 0; initialize the group counter $numGroups = 1$; initialize the scene counter $numScenes = 1$.
  2. If S is empty, quit; otherwise get the next shot. Denote this shot as shot i.
  3. Test if shot i can be merged into an existing group:
     (a) Compute the similarities between the current shot and existing groups: call the procedure $findGroupSim()$.
     (b) Find the maximum group similarity:

$$maxGroupSim_i = \max_g \quad GroupSim_{i,g} \quad , g = 1, ..., numGroups \tag{8.19}$$

where $GroupSim_{i,g}$ is the similarity between shot i and group g. Let the group of the maximum similarity be group g_{max}.

(c) Test if this shot can be merged into an existing group:

If $maxGroupSim_i > groupThreshold$, where $groupThreshold$ is a predefined threshold:

i. Merge shot i to group g_{max}.

ii. Update the video structure: call the procedure $updateStructure()$.

iii. Goto Step 2.

otherwise:

i. Create a new group containing a single shot i. Let this group be group j.

ii. Set $numGroups = numGroups + 1$.

4. Test whether shot i can be merged with an existing scene:

(a) Calculate the similarities between the current shot i and existing scenes: call the procedure $findSceneSim()$.

(b) Find the maximum scene similarity:

$$maxSceneSim_i = \max_s \quad SceneSim_{i,s} \quad , s = 1, ..., numScenes \tag{8.20}$$

where $SceneSim_{i,s}$ is the similarity between shot i and scene s. Let the scene of the maximum similarity be scene s_{max}.

(c) Test whether shot i can be merged into an existing scene:

If $maxSceneSim_i > sceneThreshold$, where $sceneThreshold$ is a predefined threshold:

i. Merge shot i with scene s_{max}.

otherwise:

i. Create a new scene containing a single shot i and a single group j.

ii. Set $numScenes = numScenes + 1$.

5. Goto Step 2.

The input to the algorithm is an unstructured video stream while the output is a structured video consisting of scenes, groups, shots, and key frames, on which the construction of video ToC is based (Figure 8.1).

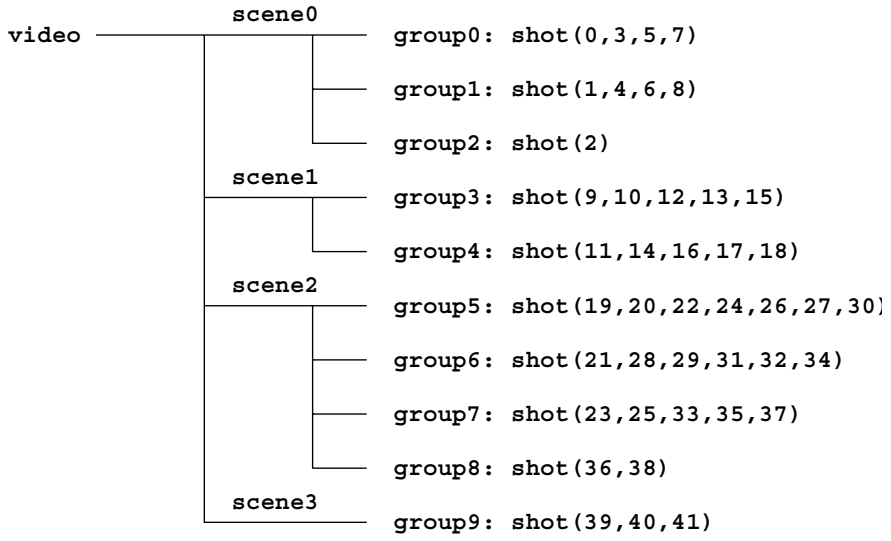


Figure 8.1 An example video ToC

findGroupSim

- Input: Current shot and group structure.
- Output: Similarity between current shot and existing groups.
- Procedure:
  1. Denote current shot as shot i.
  2. Calculate the similarities between shot i and existing groups:

$$GroupSim_{i,g} = ShotSim_{i,g_{last}} \quad , g = 1, ..., numGroups \tag{8.21}$$

where g is the index for groups and g_{last} is the last (most recent) shot in group g. That is, the similarity between current shot and a group is the similarity between the current shot and the most recent shot in the group. The reason of choosing the most recent shot to represent the whole group is that all the shots in the same group

are visually similar and the most recent shot has the largest *temporal attraction* to the current shot.

3. Return.

findSceneSim

- Input: Current shot, group structure and scene structure.
- Output: Similarity between current shot and existing scenes.
- Procedure:
  1. Denote current shot as shot i.
  2. Calculate the similarity between shot i and existing scenes:

$$SceneSim_{i,s} = \frac{1}{numGroups_s} \sum_{g}^{numGroups_s} GroupSim_{i,g} \tag{8.22}$$

  where s is the index for scenes; $numGroups_s$ is the number of groups in scene s; and $GroupSim_{i,g}$ is the similarity between current shot i and g^{th} group in scene s. That is, the similarity between current shot and a scene is the average of similarities between current shot and all the groups in the scene.

  3. Return.

updateStructure

- Input: Current shot, group structure, and scene structure.
- Output: An updated version of group structure and scene structure.
- Procedure:
  1. Denote current shot as shot i and the group having the largest similarity to shot i as group g_{max}. That is, shot i belongs to group g_{max}.
  2. Define two shots *top* and *bottom*, where *top* is the second to the last shot in group g_{max} and *bottom* is the last shot in group g_{max} (i.e., current shot).

3. For any group g, if any of its shots ($shot\ g_j$) satisfies the following condition

$$top < shot\ g_j < bottom \tag{8.23}$$

merge the scene that group g belongs to into the scene to which group g_{max} belongs. That is, if a scene contains a shot that is interlaced with the current scene, merge the two scenes. This is illustrated in Figure 8.2 (shot i = shot 4, $g_{max} = 0$, $g = 1$, top = shot 1, and $bottom$ = shot 4).

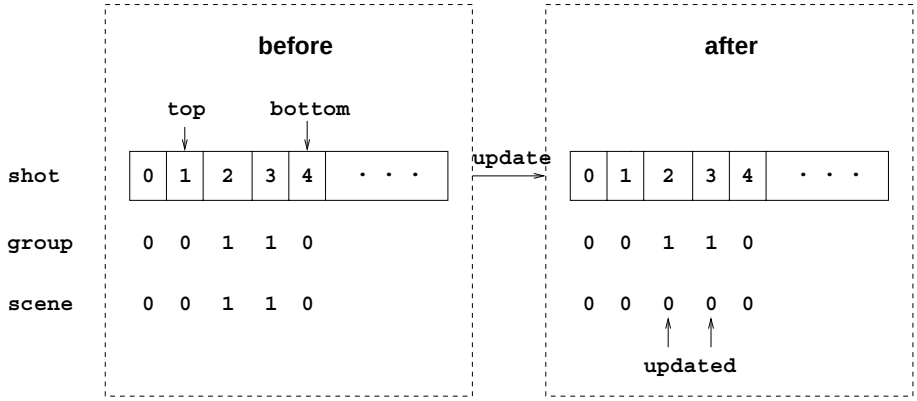


Figure 8.2 Merging scene 1 to scene 0

4. Return.

The procedure *updateStructure* is fundamentally important to scene structure construction. While $findGroupSim$ helps to group similar shots into a group and $findSceneSim$ helps to merge a shot (or a single-element group) into a scene, it is *updateStructure* that links semantically related groups into a single scene. For example, for scene 0 in Figure 8.1, while $findGroupSim$ helps to group shots 0, 3, 5, 7 into group 0; and $findSceneSim$ helps to group shot 2 to scene 0, it is *updateStructure* that links the three groups into one unified scene.

8.2.5 Comparison with existing approaches

Scene structure analysis has been previously explored in [192, 171] and in [196]. To simplify the notations, we will refer the approach described in [192, 171] as approach A and the approach

described in [196] as approach B. Comparing approaches A and B with our proposed approach, some observations can be made.

- *Temporal continuity*: In approach A, a time-window of width T is used in the *time-constrained clustering*. Similarly, in approach B, a search window of eight shots long is used when calculating the shot similarities. While this "window" approach is a big advance from the ordinary clustering in video analysis, it has the problem of discontinuity in clustering. For example, if the frame difference between two shots is $T-1$, then the similarity between these two shots is kept unchanged. But if these two shots are a bit further apart from each other, making the frame difference to be $T+1$, the similarity between these shots is cleared to 0. This discontinuity may potentially cause incorrect clustering and may make the clustering results sensitive to the window width. To overcome this discontinuity problem in our proposed approach, we introduce the concept of *temporal attraction*, which is a continuous and decreasing function of frame difference (Equations (8.7)-(8.11)). Temporal attraction effectively models the importance of time locality and does not cause any discontinuity in grouping.

- *Direct merge to a scene*: In many cases, the current shot may not be sufficiently similar to any group in a scene; thus it could not be directly merged to the scene. However, it may be similar to a certain degree to most of the groups in a scene. For example, a camera shoots three sets of shots of a person from three angles: 30, 60, and 45 degrees. Obviously, the three sets of shots will form three groups. Suppose the first two groups have already been formed and the current shot is a shot in group 3. Although the current shot may not be very similar to either group 1 or group 2, it is to some extent similar to both group 1 and group 2, and all the three groups are semantically related. This situation occurs quite often in video shooting. Approaches A and B will not be effective in handling this, they only compare the current shot to the individual groups, but not to the scene as a whole. In the proposed approach, besides calculating the similarities between current shot and groups (Step 3 in the main procedure), we also calculate the

similarities between current shot and the scene that consists of multiple groups (Step 4 in the main procedure). This added functionality effectively takes into account the above example situation and merges all the 3 groups into a unified scene.

- *On-line processing*: Another advantage of our proposed approach over approaches A and B is that the proposed approach is designed to process the video *on-line*. As the shot sequence S is coming in, the proposed approach dynamically constructs the scene structure. In approaches A and B, however, only when all the shots of the video are available can the algorithms proceed.

8.3 Determination of the Parameters

There are four parameters in the proposed video ToC construction algorithm: W_C, W_A, $groupThreshold$, and $sceneThreshold$. For an algorithm to be of practical use, all the parameters must be determined either automatically by the algorithm itself or easily by the user. In our proposed algorithm, Gaussian normalization is used in determining the four parameters. Specifically, W_C and W_A are determined automatically by the algorithm, and $groupThreshold$ and $sceneThreshold$ are determined by the user's interaction.

8.3.1 Gaussian normalization

In Equation (8.18), we combine color histogram similarity and activity similarity to form the overall shot similarity. Because the color histogram feature and activity feature are from two totally different physical domains, it would be meaningless to combine them without first normalizing them. The Gaussian normalization process ensures that entities from different domains are normalized to the same dynamic range. The normalization procedure is described as follows:

findMeanAndStddev

- Input: Video shot sequence, $S = \{shot\ 0, ..., shot\ i\}$ and a feature F associated with the shots. For example, the feature F can be either color histogram feature or activity feature.
- Output: The mean μ and standard deviation σ of this feature F for this video.
- Procedure:
  1. If S is not empty, get the next shot; otherwise goto 3.
  2. Denote current shot as shot i.
     (a) Compute the similarity in terms of F between shot i and shot i', $i' = i - MULTIPLE, ..., i-1$. Note that only the similarities of the previous $MULTIPLE$ shots need to be calculated, because shots outside $MULTIPLE$ have zero *temporal attraction* to the current shot.
     (b) Store the calculated similarity values in an array A_s.
     (c) Goto step 1.
  3. Let N_A be the number of entries in the array A_s. Consider this array as a sequence of Gaussian variables and compute the mean μ_{A_s} and standard deviation σ_{A_s} of the sequence.

The means and standard deviations for color histogram and activity measure are first calculated (denoted as μ_C, σ_C, μ_A, and σ_A) by the above normalization procedure before the scene construction procedure is applied in Section 8.2. During the scene construction procedure, μ_C, σ_C, μ_A, and σ_A are used to convert the raw similarity values to normalized ones. That is, Step 3 in Section 8.2 (Equation (8.18)) is modified to:

3. Calculate the overall shot similarity:

$$ShotSim_{i,j} = W_C * ShotColorSim'_{i,j} + W_A * ShotActSim'_{i,j} \quad (8.24)$$

$$ShotColorSim'_{i,j} = \frac{ShotColorSim_{i,j} - \mu_C}{\sigma_C} \quad (8.25)$$

$$ShotActSim'_{i,j} = \frac{ShotActSim_{i,j} - \mu_A}{\sigma_A} \quad (8.26)$$

where W_C and W_A are appropriate weights for color and activity measures; $ShotColorSim_{i,j}$ and $ShotActSim_{i,j}$ are the raw similarity values. The above procedure converts the raw similarities into similarities obeying the normal distribution of $N(0,1)$. Being of the same distribution, the normalized color histogram similarity and the normalized activity similarity can be meaningfully combined into an overall similarity. How to determine the appropriate values for W_C and W_A is discussed in the next sub-section.

8.3.2 Determining W_C and W_A

After the Gaussian normalization procedure, the raw similarities of both color histogram and activity measure are brought into the same dynamic range. That is, the normalized similarities of the color histogram feature and the activity feature contribute equally to the overall similarity. To reflect the relative importance of each feature, different weights are associated with the features.

The relative "importance" of a feature can be estimated from the statistics of its feature array A_s. For example, if all the elements in A_s are of similar value, then this particular feature has little discriminating power and should receive low weight. On the other hand, if the elements in A_s demonstrate variation, then the feature has good discriminating power and should receive high weight. Based on this intuition, the standard deviation of the feature array A_s furnishes a good estimation of the feature's importance (weight). In our case, W_C and W_A can be automatically determined as follows:

$$W_C = \frac{\sigma_C}{\sigma_C + \sigma_A} \tag{8.27}$$

$$W_A = \frac{\sigma_A}{\sigma_C + \sigma_A} \tag{8.28}$$

where σ_C and σ_A are obtained from the procedure **findMeanAndStddev**.

8.3.3 Determining *groupThreshold* and *sceneThreshold*

The *groupThreshold* and *sceneThreshold* are two important parameters in the proposed algorithm. The determination of these two thresholds would be difficult and time consuming if the shot similarities were not normalized, because in that case the thresholds are

- *Feature dependent*: Different features (color histogram versus activity measure) may require different thresholds.
- *Case dependent*: Different videos may require different thresholds.

But after the Gaussian normalization procedure, the similarity distribution of any feature for any video is normalized to the Gaussian $N(0,1)$ distribution, making the determination of thresholds much easier. The Gaussian $N(0,1)$ distribution is plotted in Figure 8.3.

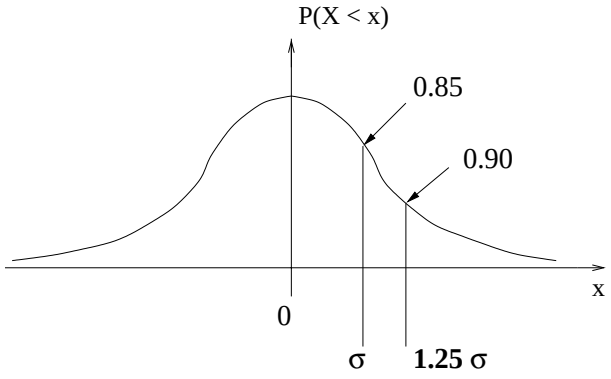


Figure 8.3 The Gaussian N(0,1) distribution

A learning process can be designed to find appropriate values for the two thresholds. Because any feature's similarity in any video is mapped to the same distribution, the training feature and video can be arbitrary. During the training, a human manually adjusts the two thresholds to obtain good video scene structure. This learning process needs to be done only once. The determined two thresholds can then be used for any other features in any other videos. Experimentally we find that $groupThreshold = \sigma = 1$ and $sceneThreshold = 0$ give good scene structure. This set of thresholds are used through out all the experiments reported in Section 8.4. Note that $P(X < x|x = groupThreshold = 1.25\sigma) = 0.90$ and $P(X < x|x = sceneThreshold = 1\sigma) = 0.85$. These two probabilities indicate that

- only if a shot is very similar to a group (better than 90% of all the shots) can it be merged to the group, and
- when a shot is similar to some extent (better than 85% of all the shots) to all the groups in a scene, it is considered similar to the scene.

These probabilities match the physical meaning of the two thresholds, as discussed in Section 8.2.

8.4 Experimental Results

Our video ToC construction system is implemented on Sun SPARC workstations, with the processing module written in C and interface module written in Java. We are currently expanding the system to deal with video streams that come from the Internet. In all the experiments reported in this section, the video streams are MPEG compressed, with the digitization rate equal to 30 frames/s. To validate the effectiveness of the proposed approach, representatives of various movie types are tested. Specifically, *The Bridges in Madison County (BMC)* (romantic-slow), *Pretty Woman (PW)* (romantic-fast), *Grease (GR)* (music), *The Mask (MS)* (comedy), *Star Trek (ST)* (science fiction-slow), *Star War (SW)* (science fiction-fast), and *Total Recall (TR)* (action) are used in our experiments. Each video clip lasts about 10-20 min. The experimental results are shown in Table 8.1, where "detected scenes" means the number of scenes detected by the algorithm; "false negatives" indicates the number of scenes missed by the algorithm; and "false positives" indicates the number of scenes detected by the algorithm that are actually not scenes.

Because *scene* is a semantic-level concept, the ground truth of scene boundary is not always concrete, and this might be why the authors of [192, 171] and [196] do not include the two columns of "false negative" and "false positive" in their experimental result tables. However, in order to judge the effectiveness of a video ToC construction approach, we believe it is useful to include those two columns. Although *scene* is a semantic concept, relative agreement can be reached among different people. The ground truth of the scene boundaries for the test

Table 8.1 Scene structure construction results.

movie name	frames	shots	groups	detected scenes	false negatives	false positives
BMC	21717	133	27	5	0	0
PW	27951	186	25	7	0	0
GR	14293	84	13	6	1	0
MS	35817	195	28	12	1	2
ST	18362	77	10	6	0	0
SW	23260	180	31	21	1	10
TR	35154	329	65	21	1	2

video sequences are obtained from a group of people who are invited to judge the algorithm's performance. The "ground truth" is the consensus of these people.

From the results in Table 8.1, some observations can be made:

- The proposed scene construction approach achieves good results in most movie types.
- The proposed approach seldom misses a scene boundary, but tends to oversegment the video. That is, the "false positive" occurs more often than "false negative." This situation is expected for most of the automated video analysis approaches and has also been observed by other researchers [192, 171].
- The proposed approach is practical in terms of parameter determination. A single set of thresholds ($groupThreshold = \sigma = 1$ and $sceneThreshold = 0$) is used through out the experiments, and W_C and W_A are determined automatically by the algorithm itself, as described in Section 8.3.

The scene structure information in Table 8.1, together with the representative frames for each scene, leads to a semantic level video ToC to facilitate user's access to the video. Figures 8.4 and 8.5 illustrate the browsing process, enabled by the video ToC. Figure 8.4 shows a condensed ToC for a video clip, as we normally have in a long book. By looking at the representative frames and text annotation, the viewer can determine which particular portion of the video clip he is interested in. In that case, the viewer can further expand the ToC into more detailed levels, such as groups and shots. The expanded ToC is illustrated in Figure 8.5. Clicking on

the "Display" button will display the specific portion that is of interest to the viewer, without viewing the entire video.

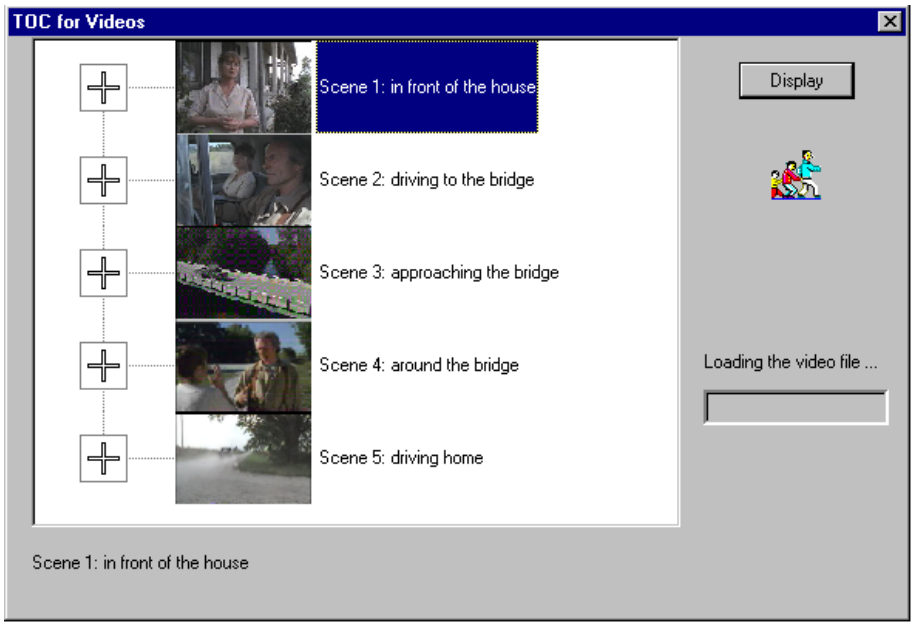


Figure 8.4 Video ToC for BMC (scene level)

The above scene-based video ToC greatly facilitates the user's access to the video. This video ToC not only provides the user with a nonlinear access to the video (in contrast to conventional linear *fast-forward* and *rewind*) but also gives the user a global "picture" of the whole story of the video. If, instead, we were using a 1D array of key frames to present the video, $2 \times 133 = 266$ frames must be presented sequentially. Because of the 1D linear display nature, even if a user can patiently browse through all the 266 frames, it is still difficult for him or her to perceive the underlying story structure.

8.5 Conclusions

With the rapid increase in digital multimedia, digital video analysis has become one of the most active research areas in the past few years. A fundamental research task in video analysis is to extract video structures, such as video ToC, to facilitate the user's access. This paper presents an effective approach to video ToC construction, in which shots are grouped

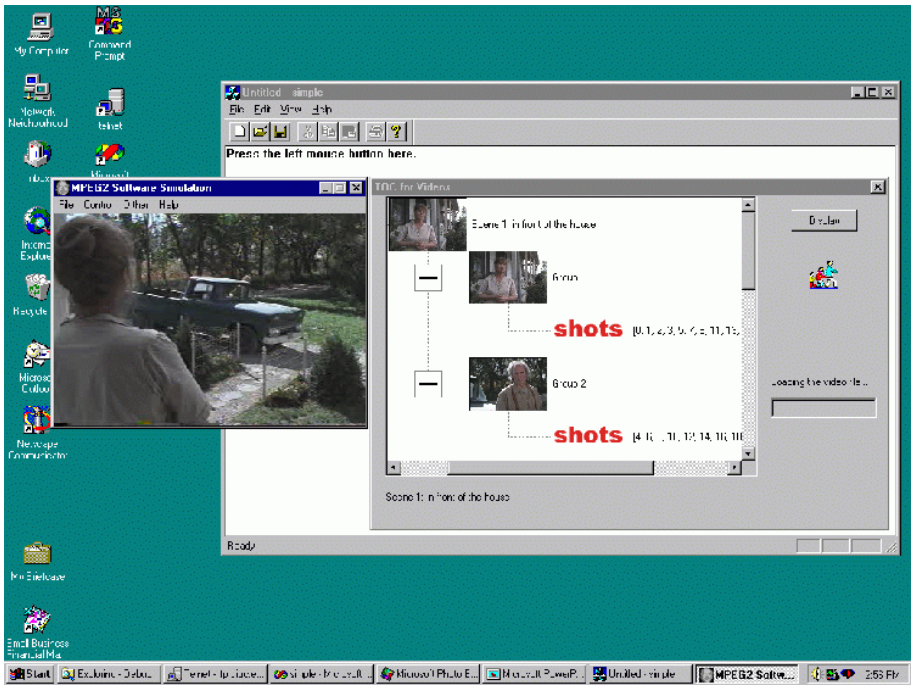


Figure 8.5 Video ToC for BMC (group level)

into semantic-related scenes. The output of the proposed algorithm provides a structured video that greatly facilitates the user's access. Experiments based on real-world movie videos validate the effectiveness of the proposed approach.

The approach proposed in this chapter provides an *open* framework for structure analysis of video – features other than the ones used in this paper can be readily incorporated for the purpose of video ToC construction. We are currently exploring integration of other visual features, as well as audio features (speech and background music) and text features (close caption), into the developed framework. We believe that an appropriate fusion of these multimodalities will result in a more semantically correct video ToC.

CHAPTER 9

VIDEO RETRIEVAL

9.1 Related Work

As we discussed in chapter 7, both ToC and Index are equally important for accessing video content. Unlike the other video representations, the mosaic representation is especially suitable for video retrieval. The three components (moving objects, backgrounds, and camera motions) are perfect candidates for video Index. After constructing such a video index, queries such as "Find me a car moving like this," "Find me a conference room having that environment," etc. can be effectively supported.

9.2 Proposed Approach

Constructing the Index for video is far more complex than constructing an index for books. For books, the form of index is fixed (e.g., keywords). For videos, the viewer's interests may cover a wide range. Depending on their knowledge and profession, they may be interested in semantic level labels (building, car, people), low-level visual features (color, texture, shape), or the camera motion effects (pan, zoom, rotation). Our current system supports the following three Index categories:

- Visual Index
- Semantic Index
- Camera motion Index

To support semantic-level- and visual-feature-based queries, frame clusters are first constructed to provide index. Our clustering algorithm is described as follows:

1. Feature extraction: Color and texture features are extracted from each frame. The color feature is 8×4 2D color histogram in HSV color space. The V component is not used because of its sensitivity to lighting conditions. The H component is quantized finer than the S component due to the psychological observation that human visual system is more sensitive to hue than to saturation. For texture feature, the input image is fed into a wavelet filter bank and is decomposed into de-correlated subbands. Each subband captures the feature of some scale and orientation of the original image. Specifically, we decompose an image into three wavelet levels; thus having 10 subbands. For each subband, the standard deviation of the wavelet coefficients is extracted. The 10 standard deviations are used as the texture representation for the image [138].

2. Global clustering: based on the features extracted from each frame, the whole video clip is grouped into clusters. The detailed description of the clustering process can be found in our previous work [188]. Note that each cluster can contain frames from multiple shots and each shot can contain multiple clusters as well. The cluster centroids are used as the visual Index and can be later labeled as semantic Index (see Chapter 10). This procedure is illustrated in Figure 9.1.

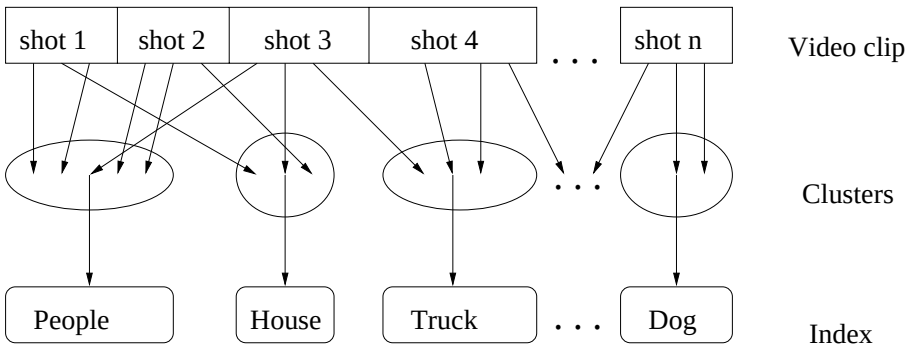


Figure 9.1 From video clip to cluster to index

After the above clustering process, the whole video clip is grouped into multiple clusters. Since color and texture features are used in the clustering process, all the entries in a given cluster are visually similar. Therefore these clusters naturally provide support for the visual queries.

In order to support semantic level queries, semantic labels need to be provided for each cluster. Instead of attempting to attack the un-solved automatic image understanding problem, semi-automatic human assistance is used in this paper. We have built interactive tools to display each cluster centroid frame to a human user and the user will label that frame. The label will then be *propagated* through the whole cluster. Because only the cluster centroid frame needs to be labeled, the interactive process is fast. For a 21,717 frame video clip (BMC), about 20 minutes is needed. After this labeling process, the clusters can support both visual and semantic queries. The specific semantic labels for BMC are people, truck, dog, tree, grass, road, bridge, house, corn field, etc.

To support camera motion queries, our lab has developed techniques to detect camera motion in the MPEG compressed domain [198]. The in-coming MPEG stream does not need to be fully decompressed. The motion vectors in the bit stream form good estimates of camera motion effects. Panning, zooming, and rotation effects can be effectively detected [198].

CHAPTER 10

A UNIFIED FRAMEWORK FOR VIDEO BROWSING AND RETRIEVAL

The previous two chapters described our video browsing and retrieval techniques separately. In this section, we will integrate them into a unified framework to enable a user to go "back and forth" between browsing and retrieval. Going from Index to ToC, a user can get the *context* where the indexed entity is located. Going from ToC to Index, a user can pin point specific queries. Figure 10.1 illustrates the unified framework.

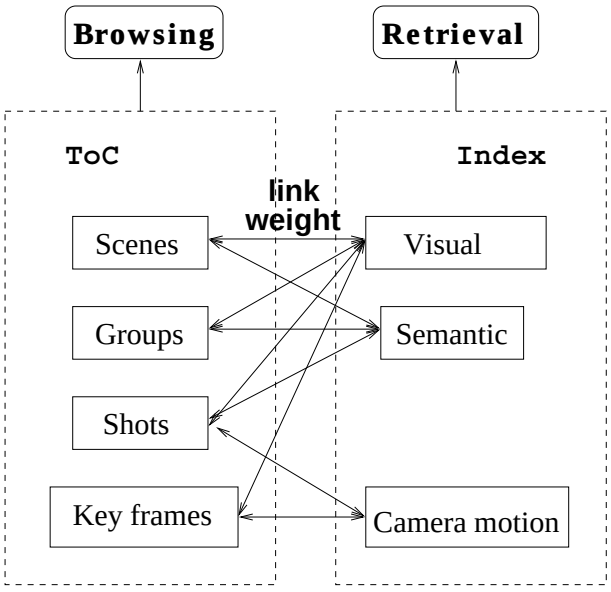


Figure 10.1 A unified framework

An essential part of the unified framework is the weighted links. The links can be established between Index entities and scenes, groups, shots, and key frames in the ToC structure. As a first step, in this paper we focus our attention on the links between Index entities and shots. Shots are the building blocks of the ToC. Other links are generalizable from the shot link.

For the link between shots and *visual Index*, we propose the following techniques. As we mentioned before, a cluster may contain frames from multiple shots. The frames from a particular shot form a subcluster. Denote this subcluster's centroid as c_{sub} and the centroid of the whole cluster as c. This is illustrated in Figure 10.2.

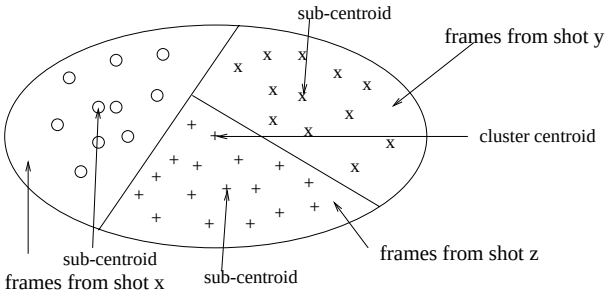


Figure 10.2 Sub-clusters

The symbol c is a representative of the whole cluster (and thus the visual Index), and c_{sub} is a representative of the frames from a given shot in this cluster. We define the similarity between the cluster centroid and subcluster centroid as the link weight between Index entity c and that shot:

$$w_v(i,j) = similarity(c_{sub}, c_j) \tag{10.1}$$

where i and j are the indices for shots and clusters, respectively; and $w_v(i,j)$ denotes the link weight between shot i and visual Index cluster c_j.

After we have defined the link weights between shots and visual Index and have labeled each cluster, we can next establish the link weights between shots and *semantic Index*. Note that multiple clusters may share the same semantic label. The link weight between a shot and a semantic Index is defined as:

$$w_s(i,k) = max_j(w_v(i,j)) \tag{10.2}$$

Table 10.1 Going from semantic, visual, camera Index to ToC.

shot id	0	2	10	12	14	31	33
w_s	0.958	0.963	0.919	0.960	0.957	0.954	0.920
shot id	16	18	20	22	24	26	28
w_v	0.922	0.877	0.920	0.909	0.894	0.901	0.907
shot id	0	1	2	3	4	5	6
w_c	0.74	0.03	0.28	0.17	0.06	0.23	0.09

where k is the index for the semantic Index entities; and j represents those clusters sharing the same semantic label k.

The link weight between shots and a *camera motion Index* (e.g., panning) is defined as:

$$w_c(i,l) = \frac{n_i}{N_i} \tag{10.3}$$

where l is the index for the camera operation Index entities; n_i the number of frames having that camera motion operation; and N_i the number of frames in shot i.

Extensive tests have been carried out over real-world video clips. The video streams are MPEG compressed, with the digitization rate equal to 30 frames/s. Table 10.1 summarizes example results over the video clip BMC. The first two rows are an example from semantic Index (e.g., truck) to ToC (shots) (also in Figure 10.4). The middle two rows are an example from visual Index (e.g., Figure 10.3) to the ToC (shots) (also in Figure 10.5). The last two rows are from camera operation Index (panning) to the ToC (shots).

Figure 10.3 Frame 2494 as a visual Index

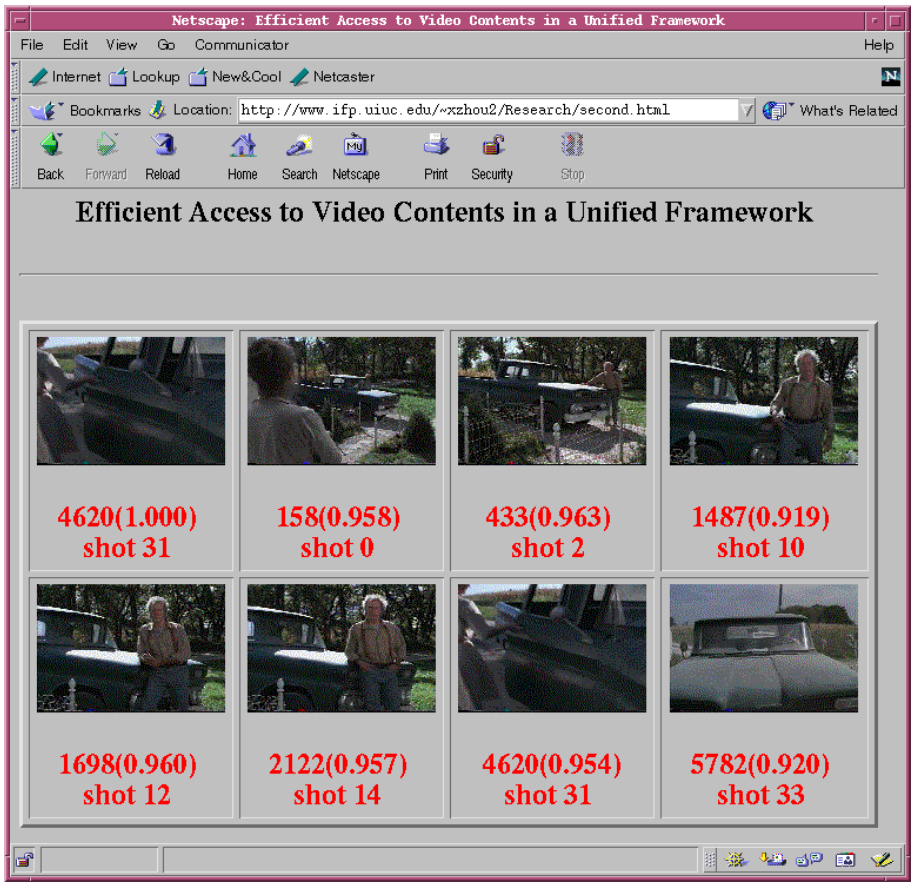


Figure 10.4 Interface for going from semantic Index to ToC

By just looking at each isolated Index alone, a user cannot understand the context. By going from Index to ToC (as in Table 10.1), a user can quickly learn when and under which circumstances (e.g., within a particular scene) that Index entity occurs. Table 10.1 summarizes how do we go from Index to ToC to find the *context*. We can also go from ToC to Index to *pin point* specific Index. Table 10.2 summarizes what are the Index entities appeared in shot 33 of the video clip BMC.

Table 10.2 Going from ToC (shots) to Index.

Index	fence	mail box	human hand	mirror	steer wheel
Weight	0.927	0.959	0.918	0.959	0.916

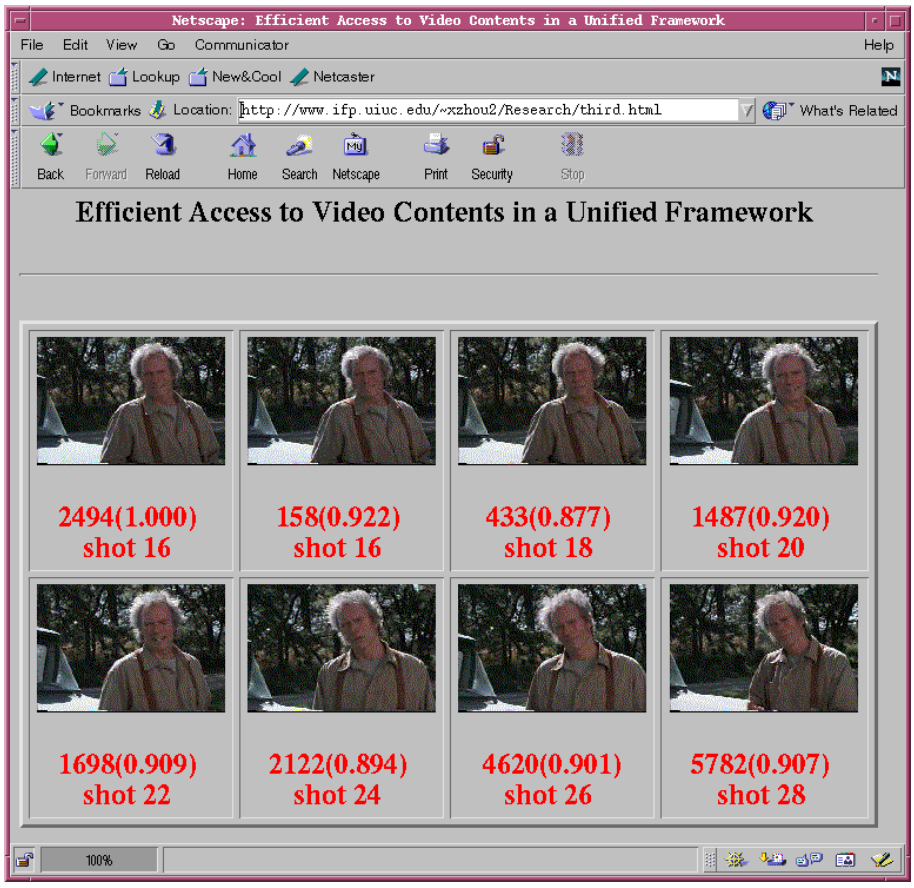


Figure 10.5 Interface for going from visual Index to ToC

For a continuous long medium type like video, a "back and forth" mechanism between browsing and retrieval is crucial. The video library users may have to browse the video first before they know what to retrieve. On the other hand, after retrieving some video objects, it will guide the users to browse the video in the correct direction. We have carried out extensive subjective tests over users from various disciplines in the university. From their feedback, this unified framework greatly facilitated their access to video contents, in both home entertainment and education.

CHAPTER 11

CONCLUSIONS AND FUTURE WORK OF VIDEO SYSTEM

The contributions of this part of the thesis are summarized as follows:

- Reviewed and discussed recent research progress in video analysis, representation, browsing, and retrieval.
- Introduced video ToC and Index and presented techniques for constructing them.
- Proposed a unified framework for video browsing and retrieval, and proposed techniques for establishing the link weights between ToC and Index.
- Based on the test results, the proposed unified framework greatly facilitated users access to video contents.

We are aware that video is not just a visual medium. It contains text and audio information, in addition to visual information, thus, "true" multimedia. Multimodel and multimedia processing is always more reliable and robust than a single medium. We are currently investigating techniques in integrating close-captioning and audio track information into our algorithm to enhance the construction of ToC, Index, and link weights.

Visual content, such as color, texture, shape, and motion vectors, are normally low-level features. On the other hand, the keywords and key concepts extracted from close-caption data are normally high-level features. By combining the two, better video representation and retrieval results can be achieved.

APPENDIX A

NORMALIZATION

One thing that we did not go into details about is the normalization process. If you look at either of Equation (4.3) or Equation (4.51), we note that if we do not make sure that the individual distance values at the representation level are in the same dynamic range, then it would be meaningless to combine them at a higher level. This is because one individual representation level distance value may overshadow the others just because its magnitude is large. For the same reason, when we calculate $d(\vec{r}_i)$'s, the vector components, r_{ik}'s, should also be normalized before applying computing the distance value. We refer the normalization of $\vec{r}_{ik}$'s as intranormalization and the normalization of $d(\vec{r}_i)$'s as internormalization [156].

A.1 Intranormalization

This normalization process puts equal emphasis on each component, r_{ik}, within a representation vector $\vec{r}_i$. To see the importance of this, note that different components within a vector may be of totally different physical quantities. Their magnitudes can vary drastically, thereby biasing the distance measure.

Assume there are M images in the database, let m be the image id index, and let $\mathcal{R}$ be a matrix obtained by stacking representation vectors $\vec{r}_{mi}$'s into each row of $\mathcal{R}_i$. We therefore have a $M \times K_i$ matrix

$$\mathcal{R}_i = [\vec{r}_{mi}^{T}] = [\mathcal{R}_{imk}],\ m = 1, ..., M,\ k = 1, ..., K_i \tag{A.1}$$

where $\mathcal{R}_{imk}$ is the m, k^{th} entry in matrix $\mathcal{R}$.

Now, the k^{th} column of matrix $\mathcal{R}_i$ is a length-M sequence. We denote this sequence as $\mathcal{R}_{ik}$. Our goal is to normalize the entries in each column to the same range so as to ensure that each individual component receives equal emphasis when calculating the distance between two vectors. One way of normalizing the sequence $\mathcal{R}_{ik}$ is to find the maximum and minimum values of $\mathcal{R}_{ik}$ and normalize the sequence to [0, 1] as follows:

$$\mathcal{R}_{imk} = \frac{\mathcal{R}_{imk} - min_{ik}}{max_{ik} - min_{ik}}, \tag{A.2}$$

where min_{ik} and max_{ik} refer to the smallest and the biggest values in the sequence $\mathcal{R}_{ik}$. Although simple, this is not a desirable normalization procedure. We will examine a sequence {1.0, 1.1, 1.2, 1.3, 100.0}. If we use Equation (A.2) to normalize the sequence, most of the [0, 1] range will be taken away by a single entry 100.0, and most of the information in {1.0, 1.1, 1.2, 1.3} is warped into a very narrow range.

A better approach is to use the Gaussian normalization. Assuming the sequence $\mathcal{R}_{ik}$ to be a Gaussian sequence, we compute the mean μ_{ik} and standard deviation σ_{ik} of the sequence. We then normalize the original sequence to a N(0,1) sequence as follows:

$$\mathcal{R}_{imk} = \frac{\mathcal{R}_{imk} - \mu_{ik}}{\sigma_{ik}} \tag{A.3}$$

It is easy to prove that after the normalization according to Equation (A.3), the probability of an entry's value being in the range of [-1, 1] is 68%. If we use $3\sigma_k$ in the denominator, according to the 3-σ rule, the probability of an entry's value being in the range of [-1, 1] is approximately 99%. In practice, we can consider all the entry values to be within the range of [-1, 1] by mapping the out-of-range values to either -1 or 1. The advantage of this normalization process over Equation (A.2) is that the presence of a few abnormally large or small values, such as the 100.0 entry in the example sequence, *does not* bias the importance of a component r_{ik} in computing the distance between vectors.

A.2 Internormalization

Intranormalization procedure ensures the equal emphasis of each component r_{ik} within a representation vector r_i. On the other hand, the internormalization procedure ensures equal emphasis of each individual distance value $d(\vec{r}_i)$ within the overall distance value d.

Depending on the distance measure $\Psi_i()$ used, the values of $d(\vec{r}_i)$'s can be of quite different dynamic ranges. In order to ensure that no single $d(\vec{r}_i)$ will overshadow the others only because it has a larger magnitude, inter-normalization should be applied. This procedure is summarized as follows:

1. For a particular representation i, compute the distance (in terms of this representation) between any pair of images in the image collection:

$$\begin{aligned} d_{m,n}() &= \Psi_i(\vec{r}_{mi}, \vec{r}_{ni}, W_i) \\ m, n &= 1, ..., M \\ m &\neq n \end{aligned} \tag{A.4}$$

2. Since there are M images in the collection, there are $C_2^M = \frac{M \times (M-1)}{2}$ possible distance values between any pair of images. Treat them as a data sequence and find the mean μ_i and standard deviation σ_i of the sequence. Store μ_i and σ_i in the database to be used in later normalization.

3. When a query Q is presented, compute the raw (unnormalized) distance values between $\vec{q}_i$ and the images in the database.

$$d_m(\vec{r}_i) = \Psi_i(\vec{r}_i, \vec{q}_i, W_i) \tag{A.5}$$

4. Normalize the raw distance values as follows:

$$d'_m(\vec{r}_i) = \frac{d_m(\vec{r}_i) - \mu_i}{3\sigma_i} \tag{A.6}$$

As explained in the intranormalization subsection, this Gaussian normalization procedure ensures that 99% of all the $d'_m(\vec{r}_i)$ values will be within the range of [-1, 1]. An additional

shift will guarantee that 99% of distance values are within [0, 1], if that is desired:

$$d''_m(\vec{r}_i) = \frac{d'_m(\vec{r}_i) + 1}{2} \tag{A.7}$$

After this shift, in practice, we can consider all the values to be within the range of [0, 1], since an image whose distance from the query is greater than 1 is very dissimilar and can be disregarded without affecting the retrieval results.

In the above normalization process, the first two steps are done off-line to obtain $\vec{\mu}_i$ and $\vec{\sigma}_i$. The last two steps are done on-line to convert the unnormalized value to normalized ones, by using the precalculated statistics $\vec{\mu}_i$ and $\vec{\sigma}_i$.

The above-described normalization process assumes that M is large enough, such that $\vec{\mu}_i$ and $\vec{\sigma}_i$ calculated based on C_2^M distance values approximate the true mean and standard deviation of the distribution of all possible $d_{m,n}(\vec{r}_i)$'s by the law of large numbers (LLN) [199]. This assumption is important because it ensures that we can use Equation (A.6) to normalize the distance value between an image and a query Q, where the query Q is arbitrary and may not be one of the images in the database.

After the intra- and internormalization procedures discussed above, the components r_{ik} within a vector $\vec{r}_i$, as well as $d(\vec{r}_i)$'s within the overall distance d, are of equal emphasis. This *objective* equality allows us to meaningfully associate *subjectively* unequal intra- and inter-weights for a particular query (Sections 4.2-4.4).

REFERENCES

[1] R. Jain, "Workshop report: NSF workshop on visual information management systems," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1993.

[2] R. Jain, A. Pentland, and D. Petkovic, "NSF-ARPA workshop on visual information management systems," (Cambridge, MA), June 1995.

[3] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafine, D. Lee, D. Petkovic, D. Steele, and P. Yanker, "Query by image and video content: The QBIC system," *IEEE Comput. Mag.*, vol. 28, pp. 23–32, September 1995.

[4] J. R. Bach, C. Fuller, A. Gupta, A. Hampapur, B. Horowitz, R. Humphrey, R. Jain, and C.-F. Shu, "The Virage image search engine: An open framework for image management," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, Feb 1996.

[5] J. Dowe, "Content-based retrieval in multimedia imaging," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1993.

[6] A. Pentland, R. W. Picard, and S. Sclaroff, "Photobook: Content-based manipulation of image databases," *Int. J. Comput. Vis.*, vol. 18, no. 3, pp. 233–254, 1996.

[7] T. S. Huang, S. Mehrotra, and K. Ramchandran, "Multimedia analysis and retrieval system (MARS) project," in *Proc. of 33rd Annual Clinic on Library Application of Data Processing - Digital Image Access and Retrieval*, 1996.

[8] J. R. Smith and S.-F. Chang, "Visually searching the web for content," *IEEE Multimedia Mag.*, vol. 4, pp. 12–20, Summer 1997.

[9] W. Y. Ma and B. S. Manjunath, "Netra: A toolbox for navigating large image databases," in *Proc. IEEE Int. Conf. on Image Proc.*, 1997.

[10] M. K. Mandal, T. Aboulnasr, and S. Panchanathan, "Image indexing using moments and wavelets," *IEEE Trans. on Consum. Electron.*, vol. 42, pp. 557–565, Aug 1996.

[11] "MPEG-7: Context and objectives (v.5)," *ISO/IEC JTC1/SC29/WG11 N1920, MPEG97*, Oct 1997.

[12] "Third draft of MPEG-7 requirements," *ISO/IEC JTC1/SC29/WG11 N1921, MPEG97*, Oct 1997.

[13] "MPEG-7 applications document," *ISO/IEC JTC1/SC29/WG11 N1922, MPEG97*, Oct 1997.

[14] V. N. Gudivada and J. V. Raghavan, "Special issue on content-based image retrieval systems," *IEEE Comput. Mag.*, vol. 28, pp. 18–62, Sept 1995.

[15] A. Pentland and R. Picard, "Special issue on digital libraries," *IEEE Trans. Patt. Recog. and Mach. Intell.*, vol. 18, no. 7-8, pp. 673–733, 1996.

[16] A. D. Narasimhalu, "Special section on content-based retrieval," *ACM Multimedia Sys.*, vol. 3, no. 1, pp. 1–41, 1995.

[17] B. Schatz and H. Chen, "Building large-scale digital libraries," *IEEE Comput. Mag.*, vol. 29, pp. 22–77, May 1996.

[18] W. Niblack, R. Barber, and et al., "The QBIC project: Querying images by content using color, texture and shape," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, Feb 1994.

[19] C. Faloutsos, M. Flickner, W. Niblack, D. Petkovic, W. Equitz, and R. Barber, "Efficient and effective querying by image content," IBM Research Report, Aug., 1993.

[20] T. S. Huang and Y. Rui, "Image retrieval: Past, present, and future," in *Proc. of Int. Symposium on Multimedia Information Processing*, Dec 1997.

[21] P. Aigrain, H. Zhang, and D. Petkovic, "Content-based representation and retrieval of visual media: A state-of-the-art review," *Multimedia Tools and Appl.*, vol. 3, pp. 179–202, Nov 1996.

[22] S. Mehrotra, K. Chakrabarti, M. Ortega, Y. Rui, and T. S. Huang, "Multimedia analysis and retrieval system," in *Proc. of the 3rd Int. Workshop on Information Retrieval Systems*, 1997.

[23] Y. Rui, T. S. Huang, and S. Mehrotra, "Relevance feedback techniques in interactive content-based image retrieval," in *Proc. of IS&T SPIE Storage and Retrieval of Images/Video Databases VI, EI'98*, 1998.

[24] Y. Rui, T. S. Huang, S. Mehrotra, and M. Ortega, "A relevance feedback architecture in content-based multimedia information retrieval systems," in *Proc. of IEEE Workshop on Content-based Access of Image and Video Libraries, in conjunction with IEEE CVPR '97*, 1997.

[25] C. Buckley and G. Salton, "Optimization of relevance feedback weights," in *Proc. of SIGIR'95*, 1995.

[26] G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval.* New York: McGraw-Hill Book Company, 1982.

[27] Y. Rui, K. Chakrabarti, S. Mehrotra, Y. Zhao, and T. S. Huang, "Dynamic clustering for optimal retrieval in high dimensional multimedia databases," in *TR-MARS-10-97*, 1997.

[28] M. Swain and D. Ballard, "Color indexing," *Int. J. Comput. Vis.*, vol. 7, no. 1, pp. 11–32, 1991.

[29] Y. Rui, T. S. Huang, S. Mehrotra, and M. Ortega, "Automatic matching tool selection using relevance feedback in MARS," in *Proc. of 2nd Int. Conf. on Visual Information Systems*, 1997.

[30] N. S. Chang and K. S. Fu, "A relational database system for images," Tech. Rep. TR-EE 79-28, Purdue University, May 1979.

[31] N. S. Chang and K. S. Fu, "Query-by pictorial-example," *IEEE Trans. on Software Eng.*, vol. SE-6, pp. 519–524, Nov. 1980.

[32] S.-K. Chang and T. L. Kunii, "Pictorial data-base systems," *IEEE Comput. Mag.*, vol. 14, pp. 13–21, Nov. 1981.

[33] S.-K. Chang, C. W. Yan, D. C. Dimitroff, and T. Arndt, "An intelligent image database system," *IEEE Trans. on Software Eng.*, vol. 14, pp. 681–688, May 1988.

[34] H. Tamura and N. Yokoya, "Image database systems: A survey," *Patt. Recog.*, vol. 17, no. 1, pp. 29–43, 1984.

[35] S.-K. Chang and A. Hsu, "Image information systems: Where do we go from here?," *IEEE Trans. on Knowledge and Data Eng.*, vol. 4, pp. 431–442, Oct. 1992.

[36] A. Gupta, S. Santini, and R. Jain, "In search of information in visual media," *Communications of ACM*, vol. 40, pp. 35–42, Dec 1997.

[37] C. S. McCamy, H. Marcus, and J. G. Davidson, "A color-rendition chart," *J. Applied Photographic Eng.*, vol. 2, pp. 95–99, Summer 1976.

[38] M. Miyahara, "Mathematical transform of (R,G,B) color data to Munsell (H,S,V) color data," in *Proc. SPIE Visual Communications and Image Processing*, vol. 1001, pp. 650–657, 1988.

[39] J. Wang, W.-J. Yang, and R. Acharya, "Color clustering techniques for color-content-based image retrieval from image databases," in *Proc. IEEE Conf. on Multimedia Comput. and Syss*, 1997.

[40] M. Ioka, "A method of defining the similarity of images on the basis of color information," Tech. Rep. RT-0030, IBM Research, Tokyo Research Laboratory, November 1989.

[41] M. Stricker and M. Orengo, "Similarity of color images," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1995.

[42] J. R. Smith and S.-F. Chang, "Single color extraction and image query," in *Proc. IEEE Int. Conf. on Image Proc.*, 1995.

[43] J. R. Smith and S.-F. Chang, "Tools and techniques for color image retrieval," in *IS & T/SPIE Proc. Vol.2670, Storage & Retrieval for Image and Video Databases IV*, 1995.

[44] J. R. Smith and S.-F. Chang, "Automated binary texture feature sets for image retrieval," in *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc.*, (Atlanta, GA), 1996.

[45] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Texture features for image classification," *IEEE Trans. on Sys, Man, and Cyb*, vol. SMC-3, no. 6, pp. 610–621, 1973.

[46] C. C. Gotlieb and H. E. Kreyszig, "Texture descriptors based on co-occurrence matrices," *Comput. Vis., Graphics, and Image Proc.*, vol. 51, pp. 70–86, 1990.

[47] H. Tamura, S. Mori, and T. Yamawaki, "Texture features corresponding to visual perception," *IEEE Trans. on Sys, Man, and Cyb*, vol. SMC-8, no. 6, pp. 460–473, 1978.

[48] W. Equitz and W. Niblack, "Retrieving images from a database using texture – algorithms from the QBIC system," Tech. Rep. RJ 9805, Computer Science, IBM Research Report, May 1994.

[49] M. Ortega, Y. Rui, K. Chakrabarti, S. Mehrotra, and T. S. Huang, "Supporting similarity queries in MARS," in *Proc. of ACM Conf. on Multimedia*, 1997.

[50] J. R. Smith and S.-F. Chang, "Transform features for texture classification and discrimination in large image databases," in *Proc. IEEE Int. Conf. on Image Proc.*, 1994.

[51] T. Chang and C.-C. J. Kuo, "Texture analysis and classification with tree-structured wavelet transform," *IEEE Trans. Image Proc.*, vol. 2, pp. 429–441, October 1993.

[52] A. Laine and J. Fan, "Texture classification by wavelet packet signatures," *IEEE Trans. Patt. Recog. and Mach. Intell.*, vol. 15, no. 11, pp. 1186–1191, 1993.

[53] M. H. Gross, R. Koch, L. Lippert, and A. Dreger, "Multiscale image texture analysis in wavelet spaces," in *Proc. IEEE Int. Conf. on Image Proc.*, 1994.

[54] A. Kundu and J.-L. Chen, "Texture classification using qmf bank-based subband decomposition," *Comput. Vis., Graphics, and Image Proc.*, vol. 54, pp. 369–384, September 1992.

[55] K. S. Thyagarajan, T. Nguyen, and C. Persons, "A maximum likelihood approach to texture classification using wavelet transform," in *Proc. IEEE Int. Conf. on Image Proc.*, 1994.

[56] J. Weszka, C. Dyer, and A. Rosenfield, "A comparative study of texture measures for terrain classification," *IEEE Trans. on Sys, Man, and Cyb*, vol. SMC-6, no. 4, pp. 269–285, 1976.

[57] P. P. Ohanian and R. C. Dubes, "Performance evaluation for four classes of texture features," *Patt. Recog.*, vol. 25, no. 8, pp. 819–833, 1992.

[58] G. C. Cross and A. K. Jain, "Markov random field texture models," *IEEE Trans. Patt. Recog. and Mach. Intell.*, vol. 5, pp. 25–39, 1983.

[59] A. P. Pentland, "Fractal-based description of natural scenes," *IEEE Trans. Patt. Recog. and Mach. Intell.*, vol. 6, no. 6, pp. 661–674, 1984.

[60] W. Y. Ma and B. S. Manjunath, "A comparison of wavelet transform features for texture image annotation," in *Proc. IEEE Int. Conf. on Image Proc.*, 1995.

[61] Y. Rui, A. C. She, and T. S. Huang, "Modified fourier descriptors for shape representation – a practical approach," in *Proc. of First International Workshop on Image Databases and Multi Media Search*, 1996.

[62] C. T. Zahn and R. Z. Roskies, "Fourier descriptors for plane closed curves," *IEEE Trans. on Comput.*, 1972.

[63] E. Persoon and K. S. Fu, "Shape discrimination using fourier descriptors," *IEEE Trans. Sys. Man, Cyb.*, 1977.

[64] M. K. Hu, "Visual pattern recognition by moment invariants," in *Computer Methods in Image Analysis*. Los Angeles: IEEE Computer Society, 1977.

[65] L. Yang and F. Algregtsen, "Fast computation of invariant geometric moments: A new method giving correct results," in *Proc. IEEE Int. Conf. on Image Proc.*, 1994.

[66] D. Kapur, Y. N. Lakshman, and T. Saxena, "Computing invariants using elimination methods," in *Proc. IEEE Int. Conf. on Image Proc.*, 1995.

[67] D. Copper and Z. Lei, "On representation and invariant recognition of complex objects based on patches and parts," in *Spinger Lecture Notes in Computer Science series, 3D Object Representation for Computer Vision*. M. Hebert, J. Ponce, T. Boult, A. Gross, Eds., New York: Springer, 1995, pp.139-153.

[68] Z. Lei, D. Keren, and D. B. Cooper, "Computationally fast bayesian recognition of complex objects based on mutual algebraic invariants," in *Proc. IEEE Int. Conf. on Image Proc.*, 1995.

[69] H. G. Barrow, "Parametric correspondence and chamfer matching: Two new techniques for image matching," in *Proc. 5th Int. Joint Conf. Artificial Intelligence*, 1977.

[70] G. Borgefors, "Hierarchical chamfer matching: A parametric edge matching algorithm," *IEEE Trans. Patt. Recog. and Mach. Intell.*, vol. 10, no. 6, pp. 849–865, 1988.

[71] E. M. Arkin, L. Chew, D. Huttenlocher, K. Kedem, and J. Mitchell, "An efficiently computable metric for comparing polygonal shapes," *IEEE Trans. Patt. Recog. and Mach. Intell.*, vol. 13, March 1991.

[72] G. C.-H. Chuang and C.-C. J. Kuo, "Wavelet descriptor of planar curves: Theory and applications," *IEEE Trans. Image Proc.*, vol. 5, pp. 56–70, January 1996.

[73] B. Li and S. D. Ma, "On the relation between region and contour representation," in *Proc. IEEE Int. Conf. on Image Proc.*, 1995.

[74] B. M. Mehtre, M. Kankanhalli, and W. F. Lee, "Shape measures for content based image retrieval: A comparison," *Information Processing & Management*, vol. 33, no. 3, pp. 319–337, 1997.

[75] I. Wallace and P. Wintz, "An efficient three-dimensional aircraft recognition algorithm using normalized fourier descriptors," *Comput. Vis., Graphics, and Image Proc.*, vol. 13, pp. 99–126, 1980.

[76] I. Wallace and O. Mitchell, "Three-dimensional shape analysis using local shape descriptors," *IEEE Trans. Patt. Recog. and Mach. Intell.*, vol. PAMI-3, pp. 310–323, May 1981.

[77] G. Taubin, "Recognition and positioning of rigid objects using algebraic moment invariants," in *Proc. SPIE Vol. 1570 Geometric Methods in Computer Vision*, 1991.

[78] T. S. Chua, K.-L. Tan, and B. C. Ooi, "Fast signature-based color-spatial image retrieval," in *Proc. IEEE Conf. on Multimedia Comput. and Syss*, 1997.

[79] H. Lu, B. Ooi, and K. Tan, "Efficient image retrieval by color contents," in *Proc. of the 1994 Int. Conf. on Applications of Databases*, 1994.

[80] R. Rickman and J. Stonham, "Content-based image retrieval using colour tuple histograms," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1996.

[81] M. Stricker and A. Dimai, "Color indexing with weak spatial constraints," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1996.

[82] G. Pass, R. Zabih, and J. Miller, "Comparing images using color coherence vectors," in *Proc. ACM Conf. on Multimedia*, 1996.

[83] J. Huang, S. Kumar, M. Mitra, W.-J. Zhu, and R. Zabih, "Image indexing using color correlogram," in *Proc. IEEE Conf. on Comput. Vis. and Patt. Recog.*, 1997.

[84] M. Lybanon, S. Lea, and S. Himes, "Segmentation of diverse image types using opening and closing," in *Proc. IEEE Int. Conf. on Image Proc.*, 1994.

[85] M. Hansen and W. Higgins, "Watershed-driven relaxation labeling for image segmentation," in *Proc. IEEE Int. Conf. on Image Proc.*, 1994.

[86] X. Q. Li, Z. W. Zhao, H. D. Cheng, C. M. Huang, and R. W. Harris, "A fuzzy logic approach to image segmentation," in *Proc. IEEE Int. Conf. on Image Proc.*, 1994.

[87] T. Gevers and V. K. Kajcovski, "Image segmentation by directed region subdivision," in *Proc. IEEE Int. Conf. on Image Proc.*, 1994.

[88] A. C. She and T. S. Huang, "Segmentation of road scenes using color and fractal-based texture classification," in *Proc. IEEE Int. Conf. on Image Proc.*, (Austin), Nov. 1994.

[89] W. Y. Ma and B. S. Manjunath, "Edge flow: a framework of boundary detection and image segmentation," in *Proc. IEEE Conf. on Comput. Vis. and Patt. Recog.*, 1997.

[90] R. Samadani and C. Han, "Computer-assisted extraction of boundaries from images," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1993.

[91] D. Daneels, D. Campenhout, W. Niblack, W. Equitz, R. Barber, E. Bellon, and F. Fierens, "Interactive outlining: An improved approach using active contours," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1993.

[92] Y. Rui, A. C. She, and T. S. Huang, "Automated shape segmentation using attraction-based grouping in spatial-color-texture space," in *Proc. IEEE Int. Conf. on Image Proc.*, 1996.

[93] D. White and R. Jain, "Algorithms and strategies for similarity retrieval," in *TR VCL-96-101*, University of California, San Diego, 1996.

[94] D. White and R. Jain, "Similarity indexing: Algorithms and performance," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1996.

[95] R. Ng and A. Sedighian, "Evaluating multi-dimensional indexing structures for images transformed by principal component analysis," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1996.

[96] C. Faloutsos and K.-I. D. Lin, "Fastmap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets," in *Proc. of SIGMOD*, pp. 163–174, 1995.

[97] S. Chandrasekaran, B. S. Manjunath, Y. F. Wang, J. Winkeler, and H. Zhang, "An eigenspace update algorithm for image analysis," *Comput. Vis., Graphics, and Image Proc.*, 1997.

[98] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, ch. 6, pp. 211–249. New York: John Wiley and Sons, Inc.

[99] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, ch. 5. Englewood Cliffs, NJ: Prentice Hall, 1993.

[100] A. Guttman, "R-tree: a dynamic index structure for spatial searching," in *Proc. ACM SIGMOD*, 1984.

[101] T. Sellis, N. Roussopoulos, and C. Faloutsos, "The R+ tree: A dynamic index for multi-dimensional objects," in *Proc. 12th VLDB*, 1987.

[102] D. Greene, "An implementation and performance analysis of spatial data access," in *Proc. ACM SIGMOD*, 1989.

[103] N. Beckmann, H.-P. Kriegel, R. Schneider, and B.Seeger, "The R*-tree: an efficient and robust access method for points and rectangles," in *Proc. ACM SIGMOD*, 1990.

[104] H. Zhang and D. Zhong, "A scheme for visual feature based image retrieval," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1995.

[105] H. Tagare, "Increasing retrieval efficiency by index tree adaption," in *Proc. of IEEE Workshop on Content-based Access of Image and Video Libraries, in conjunction with IEEE CVPR '97*, 1997.

[106] M. Charikar, C. Chekur, T. Feder, and R. Motwani, "Incremental clustering and dynamic information retrieval," in *Proc. of the 29th Annual ACM Symposium on Theory of Computing*, pp. 626–635, 1997.

[107] K. Chakrabarti and S. Mehrotra, "High dimensional feature indexing using hybrid trees," in *Proc. of IEEE ICDE99*, 1999.

[108] S.-F. Chang, A. Eleftheriadis, and R. McClintock, "Next-generation content representation, creation and searching for new media applications in education," *IEEE Proceedings*, vol. 86, pp. 602–615, Sept. 1998.

[109] B. Scassellati, S. Alexopoulos, and M. Flickner, "Retrieving images by 2D shape: A comparison of computation methods with human perceptual judgments," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1994.

[110] D. Lee, R. Barber, W. Niblack, M. Flickner, J. Hafner, and D. Petkovic, "Indexing for complex queries on a query-by-content image database," in *Proc. IEEE Int. Conf. on Image Proc.*, 1994.

[111] A. Gupta and R. Jain, "Visual information retrieval," *Communications of the ACM*, vol. 40, pp. 71–79, May 1997.

[112] "Retrievalware demo page," *http://vrw.excalib.com/cgi-bin/sdk/cst/cst2.bat*, 1997.

[113] R. W. Picard and T. P. Minka, "Vision texture for annotation," *ACM Multimedia Sys.*, vol. 3, no. 1, pp. 1–11.

[114] T. P. Minka and R. W. Picard, "Interactive learning using a "society of models"," in *Proc. IEEE Conf. on Comput. Vis. and Patt. Recog.*, 1996.

[115] R. W. Picard, "Digital libraries: Meeting place for high-level and low-level vision," in *Proc. Asian Conf. on Comp. Vis.*, Dec. 1995.

[116] R. W. Picard, "Toward a visual thesaurus," in *Springer Verlag Workshops in Computing, MIRO 95*, 1995.

[117] R. W. Picard, "Computer learning of subjectivity," *Proc. ACM Computing Surveys*, vol. 27, pp. 621–623, Dec. 1995.

[118] R. W. Picard, "A society of models for video and image libraries," Tech. Rep., MIT Media Lab, April, 1994.

[119] R. W. Picard, T. P. Minka, and M. Szummer, "Modeling user subjectivity in image libraries," in *Proc. IEEE Int. Conf. on Image Proc.*, (Lausanne), Sept. 1996.

[120] F. Liu and R. W. Picard, "Periodicity, directionality, and randomness: Wold features for image modeling and retrieval," *IEEE Trans. Patt. Recog. and Mach. Intell.*, vol. 18, pp. 722–733, July 1996.

[121] J. R. Smith and S.-F. Chang, "Intelligent multimedia information retrieval, edited by Mark T. Maybury," in *Querying by Color Regions Using the VisualSEEk Content-Based Visual Query System*, 1996.

[122] J. R. Smith and S.-F. Chang, "Visualseek: A fully automated content-based image query system," in *Proc. ACM Multimedia 96*, 1996.

[123] H. Wang and S.-F. Chang, "Compressed-domain image search and appl.," Columbia University Technical Report.

[124] S.-F. Chang, "Compressed-domain content-based image and video retrieval," in *Proc. Symposium on Multimedia Communications and Video Coding*, 1995.

[125] S.-F. Chang, "Compressed-domain techniques for image/video indexing and manipulation," in *Proc. IEEE Int. Conf. on Image Proc.*, 1995.

[126] S.-F. Chang, J. R. Smith, H. J. Meng, H. Wang, and D. Zhong, "Finding images/video in large archives," *Digi. Lib. Mag.*, Feb. 1997.

[127] J. R. Smith and S.-F. Chang, "Local color and texture extraction and spatial query," in *Proc. IEEE Int. Conf. on Image Proc.*, 1996.

[128] S.-F. Chang and J. Smith, "Extracting multi-dimensional signal features for content-based visual query," in *Proc. SPIE Symposium on Visual Communications and Signal Processing*, 1995.

[129] J. R. Smith and S.-F. Chang, "Automated binary texture feature sets for image retrieval," in *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc.*, May 1996.

[130] J. R. Smith and S.-F. Chang, "Tools and techniques for color image retrieval," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1996.

[131] A. D. Alexandrov, W.Y.Ma, A. E. Abbadi, and B.S.Manjunath, "Adaptive filtering and indexing for image databases," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1995.

[132] B. S. Manjunath and W. Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Trans. Patt. Recog. and Mach. Intell.*, vol. 18, pp. 837–842, Nov. 1996.

[133] B. S. Manjunath and W. Y. Ma, "Texture features for browsing and retrieval of image data," Tech. Rep. 95-06, Univ. of California at Santa Barbara, 1995.

[134] B. S. Manjunath and W. Y. Ma, "Image indexing using a texture dictionary," in *Proc. of SPIE conference on Image Storage and Archiving System*, vol. 2606.

[135] W. Y. Ma and B. S. Manjunath, "Texture features and learning similarity," in *Proc. IEEE Conf. on Comput. Vis. and Patt. Recog.*, pp. 425–430, 1996.

[136] W. Y. Ma and B. S. Manjunath, "A pattern thesaurus for browsing large aerial photographs," Tech. Rep. 96-10, Univ. of California at Santa Barbara, 1996.

[137] S. Mehrotra, Y. Rui, M. Ortega-B., and T. S. Huang, "Supporting content-based queries over images in MARS," in *Proc. of IEEE Int. Conf. on Multimedia Computing and Systems*, 1997.

[138] Y. Rui, T. S. Huang, and S. Mehrotra, "Content-based image retrieval with relevance feedback in MARS," in *Proc. IEEE Int. Conf. on Image Proc.*, 1997.

[139] K. Hirata and T. Kato, "Query by visual example," in *Proc. of 3rd Int. Conf. on Extending Database Technology*, March 1992.

[140] C. Carson, S. Belongie, H. Greenspan, and J. Malik, "Region-based image querying," in *Proc. of IEEE Workshop on Content-based Access of Image and Video Libraries, in conjunction with IEEE CVPR '97*, 1997.

[141] H. H. Yu and W. Wolf, "Hierarchical, multi-resolution algorithms for dictionary-driven content-based image retrieval," in *Proc. IEEE Int. Conf. on Image Proc.*, 1997.

[142] C. Frankel, M. J. Swain, and V. Athitsos, "Webseer: An image search engine for the world wide web," Tech. Rep. TR-96-14, Computer Science Department, University of Chicago, 1996.

[143] V. Athitsos, M. J. Swain, and C. Frankel, "Distinguishing photographs and graphics on the world wide web," in *Proc. IEEE Workshop on Content-Based Access of Image and Video Libraries*, 1997.

[144] S. Sclaroff, L. Taycher, and M. L. Cascia, "Imagerover: A content-based image browser for the world wide web," in *Proc. IEEE Workshop on Content-Based Access of Image and Video Libraries*, 1997.

[145] D. L. Swets and J. J. Weng, "Efficient content-based image retrieval using automatic feature selection," in *Proc. IEEE Int. Conf. on Image Proc.*, 1995.

[146] Y. Gong, H. Zhang, H. Chuan, and M. Sakauchi, "An image database system with content capturing and fast image indexing abilities," in *Proc. IEEE Int. Conf. on Image Proc.*, 1994.

[147] J. K. Wu, A. D. Narasimhalu, B. M. Mehtre, C. P. Lam, and Y. J. Gao, "Core: a content-based retrieval engine for multimedia information systems," *ACM Multimedia Sys.*, vol. 3, pp. 25–41, 1995.

[148] B. Cheng, "Approaches to image retrieval based on compressed data for multimedia database systems," Ph.D. dissertation, University of New York at Buffalo, 1996.

[149] M. J. Swain, "Interactive indexing into image databases," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, vol. 1908, 1993.

[150] A. Ono, M. Amano, and M. Hakaridani, "A flexible content-based image retrieval system with combined scene descrption keyword," in *Proc. IEEE Conf. on Multimedia Comput. and Syss*, 1996.

[151] J. P. Callan, W. B. Croft, and S. M. Harding, "The inquery retrieval system," in *Proc. of 3rd Int Conf on Database and Expert System Application*, Sept 1992.

[152] J. Mao and A. K. Jain, "Texture classification and segmentation using multiresolution simultaneous autoregressive models," *Patt. Recog.*, vol. 25, no. 2, pp. 173–188, 1992.

[153] J. R. Smith and S.-F. Chang, "An image and video search engine for the world-wide web," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1997.

[154] W. M. Shaw, "Term-relevance computations and perfect retrieval performance," *Information Processing and Management*, vol. 31, no. 4, pp. 491–498, 1995.

[155] I. J. Cox, M. L. Miller, S. M. Omohundro, and P. N. Yianilos, "Target testing and the pichunter bayesian multimedia retrieval system," in *Advanced Digital Libraries Forum*, (Washington D.C.), May 1996.

[156] Y. Rui, T. S. Huang, M. Ortega, and S. Mehrotra, "Relevance feedback: A power tool in interactive content-based image retrieval," *IEEE Trans. Circ. Sys. Video Tech.*, vol. 8, pp. 644–655, Sept. 1998.

[157] R. Fagin and E. L. Wimmers, "Incorporating user preferences in multimedia queries," in *Proc. of Int. Conf. on Database Theory*, 1997.

[158] Y. Ishikawa, R. Subramanya, and C. Faloutsos, "Mindreader: Query databases through multiple examples," in *Proc. of the 24th VLDB Conference*, (New York), 1998.

[159] K. Porkaew, S. Mehrotra, and M. Ortega, "Query reformulation for content based multimedia retrieval in MARS," in *Proc. of IEEE ICMCS99*, (Florence, Italy), June 1999.

[160] M. S. Lew, K. Lempinen, and D. P. Huijsmans, "Webcrawling using sketches," Technical Report, Computer Science Department, Leiden University, The Netherlands, 1997.

[161] J. R. Smith and S.-F. Chang, "Enhancing image search engines in visual information environments," in *IEEE 1st Multimedia Signal Processing Workshop*, June 1997.

[162] S. Weibel and E. Miller, "Image description on the internet: A summary of the CNI/OCLC image metadata on the Internet workshop, September 24 - 25, 1996, Dublin, Ohio," *Digi. Lib. Mag.*, Jan. 1997.

[163] M. Beigi, A. Benitez, and S.-F. Chang, "Metaseek: A content-based meta search engine for images," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, (San Jose, CA), 1998. Demo and document: http://www.ctr.columbia.edu/metaseek.

[164] S.-F. Chang, J. R. Smith, M. Beigi, and A. Benitez, "Visual information retrieval from large distributed on-line repositories," *Communications of ACM, Special Issue on Visual Information Retrieval*, vol. 40, pp. 12–20, Dec 1997.

[165] D. Murthy and A. Zhang, "Webview: A multimedia database resource integration and search system over web," in *WebNet 97: World Conference of the WWW, Internet, and Intranet*, Oct. 1997.

[166] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing and Management*, vol. 24, no. 5, pp. 513–523, 1988.

[167] J. Allan, "Relevance feedback with too much data," in *Proc. of SIGIR'95*, 1995.

[168] R. K. Srihari, "Automatic indexing and content-based retrieval of captioned images," *IEEE Comput. Mag.*, vol. 28, pp. 49–56, Sept 1995.

[169] J. R. Smith and S.-F. Chang, "Multi-stage classification of images from features and related text," in *Proc. 4th Europe EDLOS Workshop*, (San Miniato, Italy), Aug 1997.

[170] H. Zhang, A. Kankanhalli, and S. W. Smoliar, "Automatic partitioning of full-motion video," *ACM Multimedia Sys.*, vol. 1, no. 1, pp. 1–12, 1993.

[171] R. M. Bolle, B.-L. Yeo, and M. M. Yeung, "Video query: Beyond the keywords," Technical Report, IBM Research, Oct. 17, 1996.

[172] D. Zhong, H. Zhang, and S.-F. Chang, "Clustering methods for video browsing and annotation," Tech. Rep., Columbia University, 1997.

[173] Y. Rui, T. S. Huang, and S. Mehrotra, "Exploring video structures beyond the shots," in *Proc. of IEEE Conf. Multimedia Computing and Systems*, 1998.

[174] H. Zhang, S. W. Smoliar, and J. J. Wu, "Content-based video browsing tools," in *Proc. IS&T/SPIE Conf. on Multimedia Computing and Networking*, 1995.

[175] A. Hampapur, R. Jain, and T. Weymouth, "Digital video segmentation," in *Proc. ACM Conf. on Multimedia*, 1994.

[176] R. Kasturi and R. Jain, "Dynamic vision," in *Proc. of Computer Vision: Principles*, 1991.

[177] F. Arman, A. Hsu, and M.-Y. Chiu, "Feature management for large video databases," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1993.

[178] J. Meng, Y. Juan, and S.-F. Chang, "Scene change detection in a mpeg compressed video sequence," in *Proc. SPIE Symposium on Electronic Imaging: Science & Technology-Digital Video Compression: Algorithms and Technologies*, 1995.

[179] B.-L. Yeo, "Efficient processing of compressed images and video," Ph.D. dissertation, Princeton University, 1996.

[180] R. Zabih, J. Miller, and K. Mai, "A feature-based algorithm for detecting and classifying scene breaks," in *Proc. ACM Conf. on Multimedia*, 1995.

[181] A. Nagasaka and Y. Tanaka, "Automatic video indexing and full-video search for object appearances," in *Proc. Visual Database Systems II*, 1992.

[182] D. Swanberg, C.-F. Shu, and R. Jain, "Knowledge guided parsing in video databases," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1993.

[183] H. Zhang and S. W. Smoliar, "Developing power tools for video indexing and retrieval," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1994.

[184] H. Zhang, C. Y. Low, S. W. Smoliar, and D. Zhong, "Video parsing, retrieval and browsing: An integrated and content-based solution," in *Proc. ACM Conf. on Multimedia*, 1995.

[185] M. R. Naphade, R. Mehrotra, A. M. Ferman, T. S. Huang, and A. M. Tekalp, "A high performance algorithm for shot boundary detection using multiple cues," in *Proc. IEEE Int. Conf. on Image Proc.*, (Chicago), Oct. 1998.

[186] J. S. Boreczky and L. A. Rowe, "Comparison of video shot boundary detection techniques," in *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1996.

[187] R. M. Ford, C. Robson, D. Temple, and M. Gerlach, "Metrics for scene change detection in digital video sequences," in *Proc. IEEE Conf. on Multimedia Comput. and Syss*, 1997.

[188] Y. Zhuang, Y. Rui, T. S. Huang, and S. Mehrotra, "Adaptive key frame extraction using unsupervised clustering," in *Proc. IEEE Int. Conf. on Image Proc.*, 1998.

[189] P. O. Gresle and T. S. Huang, "Gisting of video documents: A key frames selection algorithm using relative activity measure," in *Proc. the 2nd Int. Conf. on Visual Information Systems*, 1997.

[190] W. Wolf, "Key frame selection by motion analysis," in *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc.*, 1996.

[191] M. Irani and P. Anandan, "Video indexing based on mosaic representations," *Proceedings of IEEE*, vol. 86, pp. 905–921, May 1998.

[192] M. Yeung, B.-L. Yeo, and B. Liu, "Extracting story units from long programs for video browsing and navigation," in *Proc. IEEE Conf. on Multimedia Comput. and Syss*, 1996.

[193] H. Zhang, Y. Gong, S. W. Smoliar, and S. Y. Tan, "Automatic parsing of news video," in *Proc. IEEE Conf. on Multimedia Comput. and Syss*, 1994.

[194] Y. Gong, L. T. Sin, C. H. Chuan, H. Zhang, and M. Sakauchi, "Automatic parsing of tv soccer programs," in *Proc. IEEE Conf. on Multimedia Comput. and Syss*, 1995.

[195] M. Yeung, B.-L. Yeo, W. Wolf, and B. Liu, "Video browsing using clustering and scene transitions on compressed sequences," in *Proc. of Multimedia Computing and Networking*, vol. SPIE 2417, 1995.

[196] H. Aoki, S. Shimotsuji, and O. Hori, "A shot classification method of selecting effective key-frames for video browsing," in *Proc. ACM Conf. on Multimedia*, 1995.

[197] H. Zhang, J. Y. A. Wang, and Y. Altunbasak, "Content-based video retrieval and compression: A unified solution," in *Proc. IEEE Int. Conf. on Image Proc.*, 1997.

[198] J. A. Schmidt, "Object and camera parameter estimation using mpeg motion vectors," M.S. thesis, University of Illinois at Urbana-Champaign, 1998.

[199] H. Stark and J. W. Woods, *Probability, Random Processes, and Estimation Theory for Engineers*. Englewood Cliffs, NJ: Prentice-Hall, 1986.

VITA

Yong Rui received his B.S. degree from the Southeast University, P. R. China in 1991 and the M.S. degree from Tsinghua University, P. R. China in 1994, both in electrical engineering. Since 1995, he has been with Department of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign, pursuing his Ph.D. degree. Since March, 1999, he has been a researcher at Microsoft Research, Microsoft Corp., Redmond, WA.

His research interests include multimedia information retrieval, multimedia signal processing, computer vision, and artificial intelligence. He has published over 30 technical papers in these areas.

He received a Huitong University Fellowship in 1989-1990, a Guanghua University Fellowship in 1992-1993, and a CSE College of Engineering Fellowship in 1996-1998.