THREE NEW METHODS FOR COLOR AND TEXTURE BASED IMAGE MATCHING IN CONTENT-BASED IMAGE RETRIEVAL

by

Daan He

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

at

Dalhousie University
Halifax, Nova Scotia
April 2010

DALHOUSIE UNIVERSITY

FACULTY OF COMPUTER SCIENCE

The undersigned hereby certify that they have read and recommend to the Faculty of Graduate Studies for acceptance a thesis entitled "THREE NEW METHODS FOR COLOR AND TEXTURE BASED IMAGE MATCHING IN CONTENT-BASED IMAGE RETRIEVAL" by Daan He in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Dated: April 22, 2010

External Examiner: _____

Research Supervisor: _____

Examining Committee: _____

_____

_____

_____

DALHOUSIE UNIVERSITY

DATE: April 22, 2010

AUTHOR: Daan He

TITLE: THREE NEW METHODS FOR COLOR AND TEXTURE BASED IMAGE MATCHING IN CONTENT-BASED IMAGE RETRIEVAL

DEPARTMENT OR SCHOOL: Faculty of Computer Science

DEGREE: Ph.D. CONVOCATION: October YEAR: 2010

Table of Contents

List of Tables

List of Figures

Abstract

Image matching is an important and necessary process in Content-Based Image Retrieval (CBIR). We propose three new methods for image matching: the first one is based on the Local Triplet Pattern (LTP) histograms; the second one is based on the Gaussian Mixture Models (GMMs) estimated by using the Extended Mass-constraint (EMass) algorithm; and the third one is called the DCT2KL algorithm.

First, the LTP histograms are proposed to capture spatial relationships between color levels of neighboring pixels. An LTP level is extracted from each 3×3 pixel block, which is a unique number describing the color level relationship between a pixel and its neighboring pixels. Second, we consider how to represent and compare multi-dimensional color features using GMMs. GMMs are alternative methods to histograms for representing data distributions. GMMs address the high-dimensional problems from which histograms usually suffer inefficiency. In order to avoid local maxima problems in most GMM estimation algorithms, we apply the deterministic annealing method to estimate GMMs. Third, motivated by image compression algorithms, the DCT2KL method addresses the high dimensional data by using the Discrete Cosine Transform (DCT) coefficients in the YCbCr color space. The DCT coefficients are restored by partially decoding JPEG images. Assume that each DCT coefficient sequence is emitted from a memoryless source, and all these sources are independent of each other. For each target image we form a hypothesis that its DCT coefficient sequences are emitted from the same sources as the corresponding sequences in the query image. Testing these hypotheses by measuring the log-likelihoods leads to a simple yet efficient scheme that ranks each target image according to the Kullback-Leibler (KL) divergence between the empirical distribution of the DCT coefficient sequences in the query image and that in the target image.

Finally we present a scheme to combine different features and methods to boost the performance of image retrieval. Experimental results on different image data sets show that these three methods proposed above outperform the related works in literature, and the combination scheme further improves the retrieval performance.

List of Abbreviations and Symbols Used

Σ	Covariance matrix
μ	Mean
σ^2	Variance
CBIR	Content-Based Image Retrieval
DAEM	Deterministic Annealing EM
DCT	Discrete Cosine Transform
DCT2KL algorithm	Discrete Cosine Transform 2 Kullback-Leibler divergence algorithm
EM algorithm	Expectation-Maximization algorithm
ER	Error Rate
FIRE	Flexible Image Retrieval Engine
GIFT	GNU Image Finding Tool
GMM	Gaussian Mixture Model
HSV	Hue-Saturation-Value
JPEG	Joint Photographic Experts Group
KL divergence	Kullback-Leibler divergence
LBP	Local Binary Pattern
LTP	Local Triplet Pattern
MAP	Mean Average Precision

MPEG	Moving Picture Experts Group
P-R	Precision-Recall
PCA	Principal Component Analysis
RGB	Red Green Blue
SIFT	Scale-Invariant Feature Transform
UCID	Uncompressed Colour Image Database
ZRL	Zero-Run-Length

Acknowledgements

First and foremost I want to express my deepest gratitude to my supervisor, Dr. Nick Cercone. He has been encouraging and supportive during my Ph.D study. His guidance inspires me to see problems from different perspectives. It has been an honor working with him.

I would like to thank all the professors on my committee, Dr. Christian Blouin, Dr. Qigang Gao, and Dr. Norm J. Scrimger, for their time, interest, and helpful comments. They have been supportive and providing insightful suggestions since my aptitude defense at Dalhousie.

My special thanks go to Dr. Andrew K. C. Wong. I have been impressed by his passion in research for years. I am very happy to have him as my external examiner and thanks him very much for all the precious suggestions.

I would also like to thank Ms. Menen Teferra and Dr. Dennis Riordan at Dalhousie. Ms. Teferra has helped me through all sorts of complicated paperwork since I moved out of Halifax. I am also very appreciated to Dr. Riordan for his kindly support during my study.

I am happy that I have known many good friends in these years. Anand Dersingh, Hathai Tan-tangai, and Tony Abou-Assaleh give me a hand any time I ask for a favor. Xiaofen Zheng and Jiye Li have been taking care of me since I was a fresh student at Dalhousie. Since then we have been very close friends. Jiye and Claude-Guy help me with my job searching without any hesitation. Although we are in different places now, you are always in my mind.

Most of all, I am so grateful to my family, for their love and support. Jason, Zhenmei and Dake are very important to me. They accompany and encourage me no matter what happens. My mom and dad always have strong faith in me. They give me respects and understanding. My parents-in-law and husband Hui are always there for me. Without the support of my family, I would never be able to finish my work. Thank you.

Chapter 1

Introduction

A picture is worth thousands of words. In many applications, such as art collections, photographic archives, retail catalogs, medical diagnose, crime prevention, military applications, intellectual property, architectural and engineering design, and geographical information and remote sensing systems[1], it has been seen that information is more effectively conveyed by pictures or images. For example, in online retail business, products shown in images often provide more information to potential customers. However, because of the unique characteristics of images, how to effectively show, share, organize, browse, and retrieve images from large sets presents many challenges. Among these challenges, one of the important problems is to search for images of interest, i.e., image matching.

In literature, techniques for finding matches in text passages are well established [84, 59]. In contrast, image matching proves to be more difficult, specifically because of the two challenging problems. The first challenging problem is: how to describe the image that is requested? Take as an example to find an image from a collection. A simple description such as, "portrait", or "scene", is obviously not accurate enough to find a specific image. An even more specific request, such as "an image with mountains" will match many images. The second challenging problem is: how to find an image through a large image collection according to the given image description? To answer the question, one has to go through the collection and find the matching images, either manually or using computers. On one hand, going through a large collection manually is not infeasible, but certainly time consuming; on the other hand, computers are often not as effective as humans in understanding, describing, and matching images.

Research efforts have been devoted to finding solutions for these two challenging

[1] http://en.wikipedia.org/wiki/Content_based_image_retrieval, last visited on March 15, 2010

problems. In the early years (1980's), images were matched based on image annotations, or keywords [3, 4, 9, 27, 47]. Image matching by the keywords is still one of the most popular techniques, such as used by Google [2] and Yahoo [3] image search engines. Such keyword-based image retrievals are neither efficient nor effective because of the following two reasons. First, images have to be annotated manually, which is not efficient. Second, the performance of the keyword-based image matching applications depends on how well the annotations and keywords can depict the images. Image annotations are usually biased and depend on the view of annotators. Moreover, the annotations could only partially represent image content. In the view of these, we see that to date, no truly satisfactory keyword-based methods exist for efficient and effective image matching.

1.1 Introduction to Content-Based Image Retrieval

In order to perform efficient and effective image retrieval, image retrieval turns to techniques based on image content, which is also known as Content-Based Image Retrieval (CBIR) [14, 46, 85, 100, 106]. Central to CBIR is image matching, where two images are matched by their contents. Image features, such as color, texture, and shape features, are used to match images.

Current CBIR systems are available either as commercial products, research prototypes, or demos [46, 100]. Kherfi et al. [100] provide a comprehensive survey on system design and technical achievements of current image retrieval systems, including Virage [2], Blobworld [6], Flexible Image Retrieval Engine (FIRE) [17], RetrievalWare [20], WISE [24], Netra [58], QBIC(Query By Image Content) [62], MARS [67], Photobook [71], and VisualSEEk [88], and GNU Image Finding Tool (GIFT)[4] etc. According to Kherfi et al. [100], image retrieval systems usually have one or more of the following features: random browsing, searching by example, searching by sketch, searching by text, and navigating with customized image categories. Barnard et al. [46] survey a number of prototypes for image retrieval on the web, including ImageScape [51], WebMars [66], ImageRover [82], WebSeek [88], PicToSeek [88], WebSeer [94], and WISE [103]. These prototypes usually combine text-based queries with

[2]http://images.google.ca/, last visited on March 5, 2010.
[3]http://images.search.yahoo.com/, last visited on March 5, 2010.
[4]http://www.gnu.org/software/gift/, last visited on March 5, 2010.

content-based queries.

In this thesis, we work on the image matching methods for the *searching by example* retrieval systems. In these systems, one or multiple query images are given as examples of images to be retrieved. The purpose of image matching is then to find images similar to the query images. Image matching encompasses two basic steps: image indexing and similarity comparison. Image indexing, also known as feature extraction, extracts features from images. These features are stored in a feature database for further retrieval applications. In order to index images better, one or multiple features are usually extracted from the images. The similarity between two images is then measured by the similarity between these image features. With these two steps, we are able to search for similar images to query images. Thus the retrieval performance heavily depends on how well these features can represent the images.

1.2 Motivations and Contributions

How well a feature represents an image is difficult to determine in general. In order to improve the performance of image matching from this perspective, current work is engaged in finding solutions for the following sub-questions.

1. What features to use for representing an image?
2. How to represent these features?
3. How to measure the similarities among these features?

In this thesis, we answer these three questions by presenting the following three methods: the first one is based on the Local Triplet Pattern (LTP) [37]; the second one is based on the Gaussian Mixture Models (GMMs) [28, 43, 98, 99] estimated by the EMass algorithm [38]; and the third one is the DCT2KL algorithm, which is an efficient retrieval scheme in the Discrete Cosine Transform (DCT) domain by hypothesis testing [39]. These three methods and their roles in an image retrieval system are shown in Figure 1.1.

The LTP histograms are pattern features for indexing images. GMMs provide a simplified and efficient representation for high-dimensional image features. We propose a new algorithm (EMass) to estimate GMMs from image data. The DCT2KL method selects the DCT features first, and derives the feature distance measure by hypothesis testing. We will discuss how to design and implement these proposed new

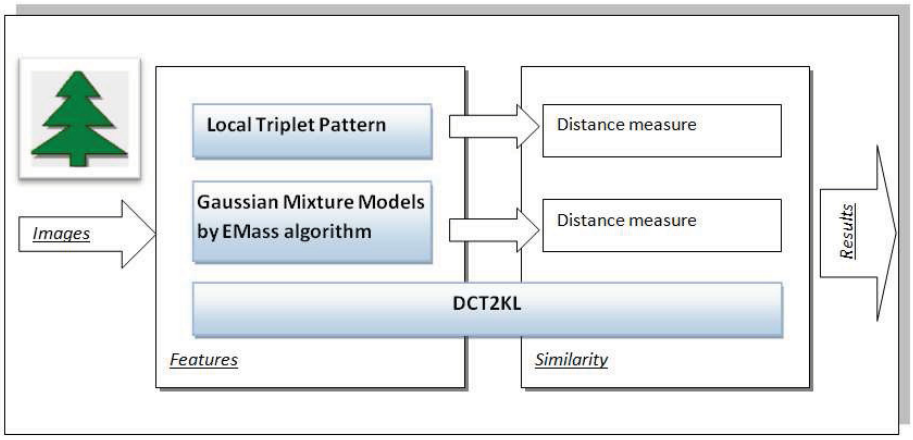

Figure 1.1: The proposed three methods and their roles in an image retrieval system.

methods, and also evaluate them by comparing our methods with state-of-the-art work.

We first introduce a pattern based image feature (i.e., Local Triplet Pattern) for color and texture images matching. The LTP feature of an image is a histogram which contains spatial information among neighboring pixels in the image. An LTP level is extracted from each 3×3 pixel block, which is a unique number describing the relationships between the color levels of a pixel and its neighboring pixels. The color levels of the eight surrounding pixels are compared to the color level of the center pixel. The comparison result of two color levels is represented by a triplet-code. Each of the triplet codes represents the three conditions: the color level of a neighboring pixel is smaller than, equal to, or larger than the color level of the center pixel. The eight triplet codes from the eight surrounding pixels are then transformed to an LTP level. We also consider extracting the LTP from a quantized color space and at different pattern length according to the application needs.

In the next step, we consider how to represent and compare multi-dimensional features by using GMMs. GMMs are alternative methods to histograms for representing data distributions. Histograms are well known for their advantages including rotation invariance, low calculation load, and so on. GMMs maintain the rotation invariance property. Moreover, GMMs address the high-dimensional problems from

which histograms usually suffer inefficiencies. To estimate GMMs from data, the Expectation-Maximization (EM) algorithm [15] is often used. To avoid local maxima problems in the EM algorithm, we apply the deterministic annealing method and propose our EMass algorithm to estimate GMMs. The objective is to minimize the overall distortion of the GMM given the data under certain constraints. We compare the EMass algorithm with the EM algorithm, and the related deterministic annealing EM (DAEM) [97] algorithm. Results show that our EMass algorithm is able to estimate GMMs more accurately and stably than the other two algorithms. When applying the algorithms for estimating GMMs from image color features for image retrieval, the EMass algorithm achieves higher precision than the other two algorithms.

We present an efficient image retrieval scheme (DCT2KL) in the DCT-domain by hypothesis testing. Motivated by image compression algorithms, we simplify the multi-dimensional features in images by using image processing techniques and the YCbCr color space[5]. More specifically, we choose the features using the DCT coefficients in the YCbCr color space. The DCT coefficients are restored by partially decoding JPEG images. In order to further decorrelate DC coefficients from an image, a 2×2 DCT is performed on the sub-image constructed from all the DC coefficients. Assume that each DCT coefficient sequence is emitted from a memoryless source, and all these sources are independent of each other. For each target image we form a hypothesis that the DCT coefficient sequences of the query images are emitted from the same sources as the corresponding sequences in the target image. Testing these hypotheses by measuring the log-likelihoods leads to a simple yet efficient scheme that ranks each target image according to the Kullback-Leibler (KL) divergence between the empirical distribution of the DCT coefficient sequences in the query image and that in the target image. Experiments on two image data sets show that our approach achieves consistently better retrieval results than related methods in literature.

Finally, we provide a simple scheme to combine different features and methods together to boost the performance of image matching. The performance of our proposed methods is compared before and after the combination on different image data sets. In addition, we also compare our results with the benchmark results on these

[5]http://en.wikipedia.org/wiki/YCbCr, last visited on March 5, 2010.

data sets. The experimental results demonstrate the effectiveness of our methods and the combination scheme.

In summary, our contributions of this thesis include,

1. Present a new image feature, the Local Triplet Pattern.
2. Present the EMass algorithm to estimate GMMs accurately and stably from multi-dimensional data.
3. Present the DCT2KL image retrieval scheme which considers feature extraction and feature comparison together by formulating the retrieval process as a hypothesis testing problem.
4. Provide a scheme to combine different features and methods.
5. Provide detailed experimental results for each method and the combination, as well as extensive discussions on their advantages and disadvantages.

In Section 1.3, we review related work in feature extraction and comparison. In order to make comparison among different methods, the retrieval performance metrics and the image data sets for evaluation are reviewed in Section 1.4.

1.3 Image Features and Comparison Methods

In this section, we review several image features and feature comparison methods. Existing image features can be grouped into two categories, global features and local features [19]. Global features are extracted from the overall image, and are often represented in form of histograms or statistical models. The popular global features include edge histograms [12], MPEG-7 color descriptors [12], color correlograms [40], Local Binary Patterns (LBP) [65], Gabor filtering features [68], color moments [92], color histograms [93], and Tamura [95] etc. Local features [54] are extracted from selective pixel blocks and patches in images. These patches carry the most distinctive information and are chosen by different methods, such as differential of Gaussian [54], corner detection methods [36] etc. Descriptors for each patch are extracted as local features. CVPIC [80], Border-Interior classification [90], and Scale-Invariant Feature Transform (SIFT) [54] are all famous local features.

1.3.1 Global Features

Basic Color Features

Colors are the most fundamental elements in images, as all image features are extracted based on colors. In the early stage of CBIR, color levels are used directly without any transformation or preprocessing. Color histograms are first introduced by Swain et al. [93]. The histograms are one of the most basic and widely used features in image retrieval [25, 63, 69, 75, 87, 91]. It has been demonstrated to be efficient and robust features for image indexing for large image data sets. A color histogram of an image describes the frequency of each color level existing in the image in pixel domain. To extract a color histogram, an image is quantized into sets of colors if necessary. Color histograms are then defined by counting the frequency or the number of times that each quantized color level existing in the image. Swain et al. show that the color histograms are invariant to translation and rotation, and tolerant to the change of angle of view, scale and occlusion. Color histograms are often used as baseline features in image matching.

Color Features Exploring Spatial Relationship

More works have been devoted to including spatial information in images to color features. The relationships between a color level of a pixel with the color levels of the surrounding pixels are explored in different features, including the color coherence vector (CCV) [69], the correlograms [40], the autocorrelograms [40], and the Local Binary Patterns (LBP) [65].

A CCV is a color histogram with each bin partitioned into two types: either coherent or incoherent. The spatial coherence is defined by the percentage of the pixels with the color level belonging to a large uniform color area in images. Regions with uniform color levels are called significant regions. The CCVs favor images with uniform color areas. Otherwise, the CCVs perform very close to the color histograms.

A correlogram of an image describes the frequency of the occurrence of two color levels, when the two color levels are spatially located at a defined distance in the image. To extract a correlogram, an image is quantized into sets of colors, and the frequency of two color levels existing at neighboring pixels are counted. The size of a

correlogram without any quantization is too large to provide efficient image retrievals. The autocorrelogram, which only counts the frequency of two identical color levels at a given distance, is presented. Both the correlogram and autocorrelogram consider the color levels at the same distance but different orientations to be the same, according to the definition of the distance function between two color levels. The main reason to define the distance function is that the feature size might be multiplied if different orientations are separately considered in feature extraction.

The LBP is a simple method to amalgamate several orientations while maintaining a low feature size. LBP transforms the relationship between two color levels into two scales, either one is smaller than or larger (equal) than the other, which can be represented by a binary code (0 or 1). The relationship of a color level of a pixel with the eight neighboring color levels are transformed into a sequence of binary code, which is equal to an LBP level ranging from 0 to 255.

Texture Features

Features examining the pattern existence and repetition in an image are called texture features. In order to describe the texture information, second or higher order statistics are primarily used together with filtering techniques, such as wavelets or Gabor filters [68, 95]. Filtering based texture features assume that the feature distribution in one filtered subband identifies one type of texture. Hence, if an image is separated into a sufficient number of subbands, the statistical signatures extracted from all subbands are sufficient to discriminate different textures in the image.

The Tamura features [95] consist of six visual perceptions: coarseness, contrast, directionality, line-likeness, regularity, and roughness. From the experiments that test the significance of these features with respect to human perception, it was concluded that the first three features are very important [19]. Thus, histograms that describe the coarseness, contrast, and directionality are often extracted as the texture features for images.

Gabor-filtered features have been widely used for texture analysis [68]. A Gabor filter is a two dimensional filtering function which extracts the significant energy from data at a specified phase and aspect ratio. In order to extract the texture information, a bank of Gabor functions is first applied at different scales and directions

to the images. The energy for the Gabor-filtered data is represented by different methods. One method is to use the mean and standard deviation of the filtered data in different orientations and at different scales, and jointly lead to multi-dimensional vector features. Another useful representation is to quantize the energy for each filter on the bank into a number of bands. The output of the mean filter over all image regions is computed into histograms to provide a global measure of the texture characteristics of the images [89].

Texture features achieve very good performance in texture image and medical image comparison, where texture information is dominant in images. For generic images, texture features are often used together with other features [28].

Shape Features

Shape features describe the object information in images, and are often classified into two types, contour-based and region-based [53]. Contour-based shape features detect the edges of an object, organize the edges into a contour, and then extract descriptions from the contour. Edge detections methods, e.g., Canny edge detector [8], Harris edge detector [35], are applied in different applications for edge detection. The contour of the object is estimated by either the polygonal [101] or the spline approximation [26]. The descriptions of the contour are often extracted by the Fourier descriptors [26, 30, 77], or chain codes [78]. Region-based shape features segment images into different regions and extract shape features from the regions, including moment, circularity, eccentricity, rectangularity, etc [55].

The extraction of shape features is a more difficult task compared to color and texture features. The shape representation is strongly affected by the image noise, distortion, and occlusion [53]. These problems affect the precision of either the edge detection or the region segmentation, and thereof, the quality of shape features. For example, the retrieval by the color and texture features in the QBIC [62] system is more precise than using the shape features. As a result, shape features are often used for retrievals by sketch or for images with man-made objects where the edges or regions are much easier to be detected. For other types of images, shape features are used together with color and texture features.

Multidimensional Features: Model-based

In order to utilize the color and other multi-dimensional data in images, different representations other than the histograms are necessary for high dimensional data. These methods include model-based features which find statistical models to represent data, and transformation-based features which transform data in pixel domain to frequency domain, e.g. the DCT domain.

Model-based features make the assumption that the distribution of features can be represented by a generic model, e.g., Gaussian Mixture Models (GMMs). The Blobworld [6] segments images into discontinued regions by GMMs, which is estimated by the EM algorithm [15]. The features for estimating GMMs are joint color-texture-position. Each segmented region is coherent in color and texture and is represented by one Gaussian component in the GMM.

S. Jeong [44] extracts quantized histogram features from the Hue-Saturation-Value (HSV) color space[6]. The quantization is performed by training a GMM from a training set with several images. The GMM is trained by the Gaussian Mixture Vector Quantization algorithm [44]. The mean for each component is used as the codebook for quantization. Histograms are then generated in the quantized color space and used for image retrieval.

A GMM-IB [29] based image retrieval method utilizes the Information Bottleneck (IB) algorithm to estimate GMMs from images to cluster images into groups. The feature space modeled by the GMMs is jointly color-spatial, with the CIE color space[7] and the pixel 2D positions. GMMs are estimated from the joint features from each image by the EM algorithm. All images are grouped into clusters by the IB algorithm, and retrievals are converted to find the closest cluster for a query image.

Multidimensional Features: Transformation-based

Different color spaces and image processing techniques have been applied to address the multi-dimensional features. Some color spaces, e.g., the HSV color space and the YCbCr color space, show more separation among three color channels than the

[6] http://en.wikipedia.org/wiki/HSL_and_HSV, last visited on March 5, 2010.
[7] http://en.wikipedia.org/wiki/CIE_1931_color_space, last visited on March 5, 2010.

RGB color space[8]. The three channels in the HSV and YCbCr can be considered as independent of each other. Thus feature extraction and comparison are processed in each channel separately. Many features have been proposed to extract from these color spaces instead of the RGB color space.

DCT-based image features have been widely applied to image retrieval applications. Two advantages of DCT-based features are as follows. Firstly, large quantities of online images or local images in databases are in JPEG format [42], which is a DCT-based image compression algorithm. DCT coefficients can be easily and quickly reconstructed from JPEG images. Secondly, the DCT coefficients are open to combine with other techniques, such as work in [25, 83, 105, 107, 108].

The color layout [25] feature is extracted based on DCT coefficients instead of colors. The color layout first divides an image into 64 blocks (8 rows by 8 columns) with an equal size. The average color for each block is calculated. 2D DCT is applied to the 64 average colors, and the resulting DCT coefficients are the extracted layout features. Color layout feature is very sensitive to the orientation of the image, as it is actually not a real histogram. The partition into 64 blocks is too coarse that the average color cannot represent the block precisely, which decreases its discriminating capability.

Several other methods have been presented working with the DCT coefficients restored from JPEG images. Lay et al. [49] build histograms from selected group of coefficients, which are either from a 1x1, 2x2 or 3x3 block. DC and AC coefficients are counted in a same histogram. Histograms are compared by a normalized Minkowski-form (L_1) distance. They conclude that the histograms from the DC coefficient are suitable for image similar in colors, and combinations of DC with low-level AC coefficients usually yield better results for generic images.

Liu et al. [57] proposes a joint color and texture DCT feature based on the coefficients spatial and energy properties. Each 8×8 DCT block is split into four sub-blocks. The color features which are four color histograms of the mean values of the sub-blocks are constructed. The texture features is the histogram of the mean and the standard deviation of selected coefficient blocks, which contain texture information such as edges in the DCT blocks.

[8]http://en.wikipedia.org/wiki/RGB, last visited on March 5, 2010.

D. Zhong [109] proposes to extract two histograms from the DCT coefficients in 4×4 DCT blocks. The histograms are extracted from patterns defined similar to the LBP [65]. The corresponding DCT coefficients are grouped into sub-images. Coefficients in each 3×3 block are transformed into sequences of binary codes or ternary codes by comparing to predefined thresholds. Each sequence in a block is converted to a number, and histograms of the resulting numbers are generated as the features.

Feng et al. [23] analyzes the statistical features of the DCT coefficients. They provide a simple method to calculate the average DC coefficient and weighted square mean of the AC coefficients. The statistical parameters of mean and variance are quantized into 28 subspaces. According to these 28 subspaces, a histogram is constructed as the feature vector of an image.

1.3.2 Local Features

Local features are extracted from local patches in images. They have provided promising performance for object recognition. For objects in images, there are many points that carry the most distinctive information about the objects. These points are known as interesting points. To extract local features from an image, interesting points are first extracted by different methods, e.g., differential of Gaussian, Harris corner detection, etc. Unstable interesting points are removed, and descriptors are then extracted from the remaining interesting points.

It is important that the local features are robust to changes in image scale, noise, illumination and local geometric distortion, for performing reliable recognition. Scale-Invariant Feature Transform (SIFT) [54] is one of the best algorithms in computer vision to detect and describe local features in images. SIFT is able to robustly identify objects even among clutter and under partial occlusion because the SIFT descriptors are invariant to scale, orientation, affine distortion and partially invariant to illumination changes. SIFT consists of four steps: scale-space extrema detection, keypoint localization, orientation assignment, and keypoint descriptor extraction. The resulting SIFT features for an image are a group of descriptors of size 128.

Although the dimension of each descriptor in the SIFT is 128, the total size of a SIFT feature for an image is very large. For an image of size 256×256, the number of

descriptors is usually of the size of several hundreds to thousands. Some approaches improve the SIFT feature efficiency by decreasing the size of the descriptor, such as the PCA (Principal Component Analysis) SIFT features [45]. The PCA SIFT features use the PCA to downsize the SIFT descriptor from 128 to a much smaller size, i.e., 32. Like the SIFT features, the PCA SIFT features locate all keypoints by the same method. The main change is that the descriptors are processed by the PCA and represented by the gradient patch around the keypoints. Experimental results have shown that PCA SIFT features are more compact and distinctive than the SIFT features.

SURF (Speeded Up Robust Features) [6] is a robust image feature inspired by the SIFT feature. The SURF filters images with the 2D Haar wavelet responses, and uses the integral images to improve the computation in near constant time. The standard version of the SURF is several times faster than the SIFT [6].

The local features are usually designed for object recognition purposes. To incorporate into an image retrieval system, different representation methods are applied, such as, local histograms [18], local signatures [61], and global search [19]. For each representation method, local features (e.g., SIFT) are first extracted from each image. In the local histograms [18], all features are first clustered into N clusters. For each local feature, we only record the id of the cluster that the feature are closest to. Finally a size N histogram is extracted from all features in an image. The local signatures [61] cluster the features for each image separately. The mean and covariance are used for comparison between images. The global search [19] method uses efficient nearest neighbor searching algorithms to find matching descriptors. For each query image, we globally search for the top k nearest neighbor descriptors in all target images. The number of descriptors belonging to each target images is calculated and sorted. Thus, the similarity is decided by the number of descriptors that a target image has.

Global and local features have their own advantages and disadvantages. Global features are more efficient, and provide overall matching between images. Local features provide more detail information at the object level, but take more time to compute and process. In this thesis, we focus on methods to extract global features and decide on the corresponding matching schemes. The LTP histograms explore the

spatial information among neighboring pixels. The EMass algorithm estimates model-based features i.e., GMMs, from multi-dimensional data. The DCT2KL scheme uses DCT coefficients partially restored from JPEG images, and then derives the specific distance measure theoretically.

1.3.3 Feature Comparison

Feature comparison uses a measurement to calculate how similar are two features. The comparison method is chosen based on what type of representation that the features adopt. Observing the features reviewed in Section 1.3.1 and 1.3.2, features are usually represented by second-order (or higher-order) statistics, non-parametric histograms, or multi-modal density function [28]. Different distance measures are applied according to the features' different representation.

Distance Measure for Histogram-based Features

Histogram representation is often used for features [69, 75, 87, 91]. Distance measurements for histograms include L_p, cosine distance, normalized histogram intersection match [93], quadratic distance [62], Earth Mover Distance (EMD) [50], Kullback-Leibler (KL) divergence [13], and Jensen-Shannon divergence (JSD) [74]. Given two histograms H_1 and H_2 of the same size K, these distance measurements are defined as follows.

The L_p measure is defined in Equation (1.1).

$$D_{L_p}(H_1, H_2) = \left(\sum_{k=1}^{K} (H_1(k) - H_2(k))^p \right)^{\frac{1}{p}} \tag{1.1}$$

where p is often chosen as 1 or 2. The L_1 is also known as Manhattan distance and the L_2 is known as Euclidean distance. When the features are not distributions, such as jointly high-order moments, L_1 is often applied to gain the absolute distance between histograms.

The cosine distance defines the similarity between two histograms by finding the cosine of the angle between them.

$$D_{cos}(H_1, H_2) = \frac{\sum_{k=1}^{K} (H_1(k) H_2(k))}{\sqrt{\sum_{k=1}^{K} H_1(k)^2} \sqrt{\sum_{k=1}^{K} H_2(k)^2}} \tag{1.2}$$

The cosine distance is a modified normalized L_2 distance, and is often used in text mining.

The histogram intersection match is defined in Equation (1.3).

$$D_i(H_1, H_2) = \frac{\sum_{k=1}^{K} \min(H_1(k), H_2(k))}{\sum_{k=1}^{K} H_2(k)} \tag{1.3}$$

The normalized histogram intersection match reduces the effect of the pixels in the background on the feature matching.

The quadratic distance [62] is introduced by the QBIC and is defined in Equation (1.4)

$$D_{quad}(H_1, H_2) = (H_1 - H_2)' A (H_1 - H_2) \tag{1.4}$$

where A is a similarity matrix of size $K \times K$. The quadratic distance includes the correlations between color levels to the distance. The cross-correlation among colors is represented by the matrix A, which is predefined given the specific color space. For the RGB color space, the matrix is defined as:

$$a_{ij} = 1 - \frac{dl_{ij}}{\max(dl_{ij})}, 1 \leq i, j \leq K \tag{1.5}$$

where dl_{ij} is L_2 distance between the color i and j.

The Earth Mover Distance (EMD) between two histograms is the minimum cost of turning one of them into the other. Informally, if the distributions are interpreted as two different ways of piling up a certain amount of dirt, the EMD is the minimum cost of turning one pile into the other. The cost is the amount of dirt to be moved times the distance by which it is moved. To define the EMD between H_1 and H_2 is to find the best $F = \{f_{ij}, 1 \leq i, j \leq K\}$ to minimize the overall cost $W(H_1, H_2, F) = \sum_{i=1}^{K} \sum_{j=1}^{K} f_{i,j} d_{ij}$, under the constraints that

$$f_{ij} \geq 0, \sum_{i=1}^{K} \sum_{j=1}^{K} f_{ij} = 1, 1 \leq i, j \leq K \tag{1.6}$$

$$\sum_{j=1}^{K} f_{ij} \leq H_1(i), \sum_{i=1}^{K} f_{ij} \leq H_2(j) \tag{1.7}$$

With the optimal F, the EMD is given as

$$D_{EMD}(H_1, H_2) = \frac{\sum_{i=1}^{K} \sum_{j=1}^{K} f_{ij} d_{ij}}{\sum_{i=1}^{K} \sum_{j=1}^{K} f_{ij}} \tag{1.8}$$

The KL divergence is defined as

$$D_{KL}(H_1, H_2) = \sum_{k=1}^{K} H_1(k) \log \frac{H_1(k)}{H_2(k)} \tag{1.9}$$

The KL divergence is a popular method of measuring the similarity between two probability distributions. The KL divergence measures the expected numbers of bits required to encode a sequence of data generated from one distribution but using the code based on the other distribution.

Jensen-Shannon divergence (JSD) is defined as

$$D_{JS}(H_1, H_2) = \sum_{k=1}^{K} H_1(k) \log \frac{2H_1(k)}{H_1(k) + H_2(k)} + H_2(k) \log \frac{2H_2(k)}{H_1(k) + H_2(k)} \tag{1.10}$$

The JSD is based on the KL divergence. But unlike the KL divergence, the JSD is symmetric, bounded, and a true distance metric.

Distance Measure for Model-based Features

Histograms suffer heavy computational load when the features are in high dimension. Quantization [32, 44, 87] is thus applied to alleviate the high complexity in this situation. However, the optimal quantization intervals are too difficult to determine, and inappropriate quantization will decrease the histogram's discrimination capability. The model-based representation [28] is suggested to be a good balance between efficiency and effectiveness for feature representation. Features for an image are represented by a model which describes the feature distribution. As the distribution of features in an image is usually a mixture of several distributions, a mixture model is always applied, e.g., GMMs. The distance between two models is more difficult to measure than the distance between two histograms. The most accurate method is based on the Monte-Carlo simulations, which is very time-consuming. Current solutions include the measurements by the approximation of the KL divergence [28], the unscented transformation distance [28], the approximation of the EMD [41], and the approximation of the normalized L_2 distance [41]. Goldberger et al. [28] show that the unscented transformation distance is an efficient method and is very close to large sample Monte-Carlo based ground truth.

Some distance metrics are designed for specific features, such as the region-based distance measure [102] and the fuzzy feature matching method [10]. These metrics are not included here.

1.4 Performance Evaluation

In order to evaluate the effectiveness of image features for retrieval, image data sets and performance metrics are also very important. In this section, we review the image data sets and retrieval performance metric used for evaluation.

1.4.1 Benchmark Image Data Sets

In order to compare the performance of different image features for retrieval systems or applications, benchmark image data sets have been proposed in literature [19]. However, compared to all types of features, publicly available image data sets are limited for several reasons, such as copyrights, missing ground truth data, and lack of annotations. Despite these limitations, some image data sets available for research purposes remain useful. These data sets usually provide ground truth data, or group images into different categories. With such information, comparison among different features is possible. We introduce several data sets that are public for research purposes.

OUTEX The OUTEX [64] image data sets[9] provide several groups of texture images for image classification and segmentation applications. Their texture images come from three sources: the Brodatz album [7], the MIT Vision Texture database[10], and the MeasTex database [86][11]. Each texture image is cut into small images with a window size varying from 32×32 to 128×128 pixels to generate different data sets, numbered from Outex_TC_00000 to Outex_TC_000016. The size of each group varies from several hundreds to thousands. For each data set, the ground truth data contains the class label for each image. The training and testing data and the best classification results on the each data are provided as well. These data sets are very

[9] http://www.outex.oulu.fi, last visited on March 01, 2010.

[10] http://vismod.media.mit.edu/vismod/imagery/VisionTexture/vistex.html, last visited on March 01, 2010.

[11] http://www.texturesynthesis.com/meastex/meastex.html, last visited on March 01, 2010.

good resources for comparing the performance of classification by using different image texture features. Some texture images from the OUTEX data set are shown in Figure 1.2. To evaluate the performance of image retrievals, we can consider the

Figure 1.2: Example images from the OUTEX data set.

images with the same class label are positive matches (relevant). The images with different class labels are negative (not relevant).

Corel1K The Corel1K [52][12] data set consists of 1000 images from the Corel stock photo database. The 1000 images fit into 10 categories: Africa, Beach, Building, Buses, Dinosaurs, Elephants, Horses, Flowers, Mountains, and Food, with 100 images in each category. An example image from each category is shown in Figure 1.3.

For retrieval tasks, we can consider the images in one category as positive matches, and images in different categories as negative matches. In this way, the Corel1K data set can be used for evaluating features and methods for image matching.

UW The UW database[13] is created by the University of Washington with 1019 images. Images are of varying sizes, and mainly represent natural scenes from different

[12]http://wang.ist.psu.edu/docs/related/, last visited on March 01, 2010.

[13]http://www.cs.washington.edu/research/imagedatabase/groundtruth, last visited on March 01, 2010.

Figure 1.3: Example images from the Corel1k data set.

locations, such as: Australia, Cambridge, spring flowers, and Yellow Stone. Some example images are shown in Figure 1.4.

Figure 1.4: Example images from the UW data set.

All images are put into different categories, and annotated with several keywords (maximum 22 keywords, and minimum 1 keyword). On average, each image is associated with 6 keywords. The relevance between images is decided by keywords matching. Two images are relevant if they both have a common keyword. On average, each image has 53 relevant images in the data set.

UCID The Uncompressed Colour Image Database (UCID) data set [81][14] consists of 1338 uncompressed images. Images represent buildings, natural scenes, small man-made objects etc. A list of 262 queries is provided. The relevant images to each query image are listed in a ground truth file for performance evaluation. Some example

[14]http://vision.cs.aston.ac.uk/data sets/UCID/, last visited on March 01, 2010.

images are shown in Figure 1.5. The ground truth for the UCID shows that the

Figure 1.5: Example images from the UCID data set.

retrievals require information at the object level.

IAPR TC-12 The image collection of the IAPR TC-12 data set [15] consists of 20,000 still natural images which are taken from locations around the world. This data set includes pictures of different sports and actions, photographs of people, animals, cities, landscapes and many other aspects of contemporary life. Some example images are shown in Figure 1.6.

Figure 1.6: Example images from the IAPR data set.

The images have full-text annotations. This data set has been used as a benchmark data set for ImageCLEF (The CLEF Cross Language Image Retrieval Track) photo retrieval tasks [33] in 2006, 2007 and 2008. The ImageCLEF retrieval tasks provide a group of retrieval topics, and the corresponding ground truth data for the evaluation.

[15] http://www.imageclef.org/photodata, last visited on March 01, 2010.

IRMA The IRMA data set[16] consists of 12000 radiograph images in 119 categories. All images are fully annotated, and separated into a training set with 10000 images, an extended training set with 1000 images, and a testing set with 1000 images.

The IRMA is used for testing image classification applications. For retrieval applications, a similar method is adopted as to the Corel1K data set. Images are relevant if they belong to the same category, and irrelevant otherwise.

We list all data sets in Table 1.1. The size, ground truth information and distinctive characteristics for each data set are listed. We list one of the OUTEX data sets as an example.

Table 1.1: Summary of the image data sets

data set	images	queries	characteristics
OUTEX_TC_13	1,360	1,360	texture images
Corel1K	1,000	1,000	images in 10 categories
UCID	1,338	262	natural scene images with ground truth data
UW	1,109	1,109	natural scene images in different categories
IAPR TC-12	20,000	60	mixture of different images with 60 queries and results
IRMA	12,000	1000	medical images in 119 categories

In our thesis, we evaluate our proposed methods on the OUTEX, the Corel1K, the UCID, and the IAPR TC-12 image data sets. These popular data sets are used in our experiments because many related methods have been tested on them. Thus, we are able to compare our methods to the related work with their reported results on the same data sets.

The OUTEX data sets are images with textures. Benchmark results on these data sets are also reported. Thus these data sets are suitable for evaluating texture features. Our LTP features provide spatial relationship between neighboring pixels, which partially describe texture in images. Thus, the performance of the LTP features is tested on the OUTEX data sets.

Our methods are also tested by the performance of image retrievals on the Corel1K, the UCID, and the IAPR TC-12 data sets. From Table 1.1, we can see that the Corel1K, UCID, and UW data sets are of similar size, and images are in similar categories. Among them, more experimental results are reported on the Corel1K and

[16]http://ganymed.imib.rwth-aachen.de/irma/datasets_en.php, last visited on March 01, 2010.

UCID. Thus we will evaluate our proposed methods on these two data sets to make comparison with other work.

From 2006 to 2008, image retrieval tracks have been organized and performed using the IAPR data sets. We will also report the experimental results by using our methods for the IAPR TC-12 data set to compare with the reported results in the ImageCLEF retrieval tracks. The IRMA data set consists of medical images which require special techniques and features. Thus, we do not show results on this data set.

1.4.2 Performance Metric

In order to evaluate the retrieval results from the data sets, measurements are necessary to rate the performance. The basic measures are based on Precision (P) and Recall (R), which are defined in Equations (1.11) and (1.12):

$$P = \frac{\text{Number of relevant images retrieved}}{\text{Total number of images retrieved}} \tag{1.11}$$

$$R = \frac{\text{Number of relevant images retrieved}}{\text{Total number of relevant images in the dataset}} \tag{1.12}$$

The P and R values are usually shown as a P-R graph. F_β-score is a measure that combines precision and recall into one number.

$$F_\beta = (1+\beta^2)\frac{P \times R}{\beta^2 P + R} \tag{1.13}$$

F_β-score is a balanced measure for P and R. β changes the importance of precision and recall. β is often chosen as 1 or 2.

The average precision is defined as the mean precision after each relevant image is retrieved. The mean average precision (MAP) is the mean of the precision for each relevant image is retrieved. The MAP is defined in equation (1.14):

$$\text{Mean Average Precision} = \frac{\sum_{r=1}^{N}(\text{pre}(r) \times \text{rel}(r))}{\text{number of relevant documents}} \tag{1.14}$$

where r is the rank of the retrieved image, N is the total number of all retrieved images, rel$(\cdot)$ is a binary function indicating whether the retrieved image at the given rank r is relevant to the query image, and pre$(\cdot)$ is the precision at the given rank r. This measure favors the features that can retrieve relevant images earlier.

1.5 Organization of the Thesis

The remainder of this thesis is organized as follows. First we introduce the new image feature, i.e., Local Triplet Pattern (LTP), in Chapter 2. We discuss how to design the LTP features, and compare to the related image features that also include spatial information. Experimental results demonstrate that the proposed LTP histograms achieve higher retrieval precision performance on both the texture image data sets (OUTEX) and the generic image data set (Corel1K) than these related features.

In Chapter 3, we discuss how to effectively utilize and represent the multi-dimensional data, i.e., color information, for retrieval. The Gaussian Mixture Models (GMMs) are used to represent the multi-dimensional data. The EMass algorithm is applied to estimate GMMs from data. The local maxima problem in the EM algorithm is avoided by applying the deterministic annealing method. By comparing the GMMs estimated by the EM algorithm, the EMass algorithm, and the related DAEM algorithm using simulation experiments, we can see that the EMass algorithm is the most stable among these three algorithms. Retrieval results on the Corel1K data set show that the GMMs by the EMass algorithms achieve higher precision than the related work.

In Chapter 4, we continue to work on utilizing high-dimensional data in images. Motivated by image compression algorithms, we work on the YCbCr color space instead of the RGB color space, and use the DCT coefficients by partially decoding JPEG images. The DCT2KL scheme is derived directly by formulating the image retrieval process as a hypothesis testing problem. We show that the KL divergence is an optimal distance measure for the DCT coefficient features. Experimental results on both the COREL1K and UCID data sets show that our DCT2KL scheme consistently outperforms related work often by a wide margin.

We work on combining our proposed features and methods to improve the retrieval performance in Chapter 5. The performance of our combined retrieval methods are compared to the state-of-the-art work on the Corel1K and the UCID data sets. Several retrieval cases are shown to give examples of how our features positively affect the retrievals. Our methods are then applied to image retrieval tasks for the IAPR TC-12 data set. We compare the retrieval results with the reported results in the ImageCLEF 2007.

In Chapter 6, we summarize our contributions, and discuss the future work.

Chapter 2

Local Triplet Pattern Histograms

Histogram-based features commonly used in image matching range from the color histograms to more complicated histograms of local features. Histograms can be easily and quickly extracted. They are invariant to rotation, and robust to occlusion and changes of view. For these reasons, histograms are the most popular features used for image matching.

In this chapter, we propose a new histogram-based image feature: Local Triplet Pattern (LTP) [37] for image matching. The proposed LTP feature of an image is a one-dimensional histogram which describes the spatial relationships of the color levels in pixel blocks in the image. Before presenting the LTP histograms, we first introduce several well-known histogram-based features, which are the color histograms, the correlograms [40], the autocorrelograms [40], and the Local Binary Pattern (LBP) [65] histograms. After reviewing and discussing these features, we observe that the color histogram features can be improved by containing more detailed spatial information among neighboring pixels. The LTP feature is then proposed as a histogram of LTP levels which contain spatial relationships of the neighboring pixel color levels in a 3×3 pixel block. The performance of the LTP histograms is evaluated and compared to the related histogram features on both the texture and generic image data sets. Experimental results show that the LTP histograms provide consistently better performance than the related features.

2.1 Background

2.1.1 Color Histograms

Color histograms are widely used in image indexing [25, 63, 69, 75, 87, 91, 93]. A color histogram of an image describes the frequency of each color level in the image in pixel domain. To extract a histogram, an image is quantized into n sets of colors

$C = \{c_1, \ldots, c_n\}$ if necessary. A histogram H is a vector $H = (h_1, \ldots, h_n)$, with each bin $h_i (1 \leq i \leq n)$ as the frequency in color c_i. An example in Figure 2.1 shows an image, the corresponding grey-level histogram, and one-dimensional histograms in the R, G, B channels.

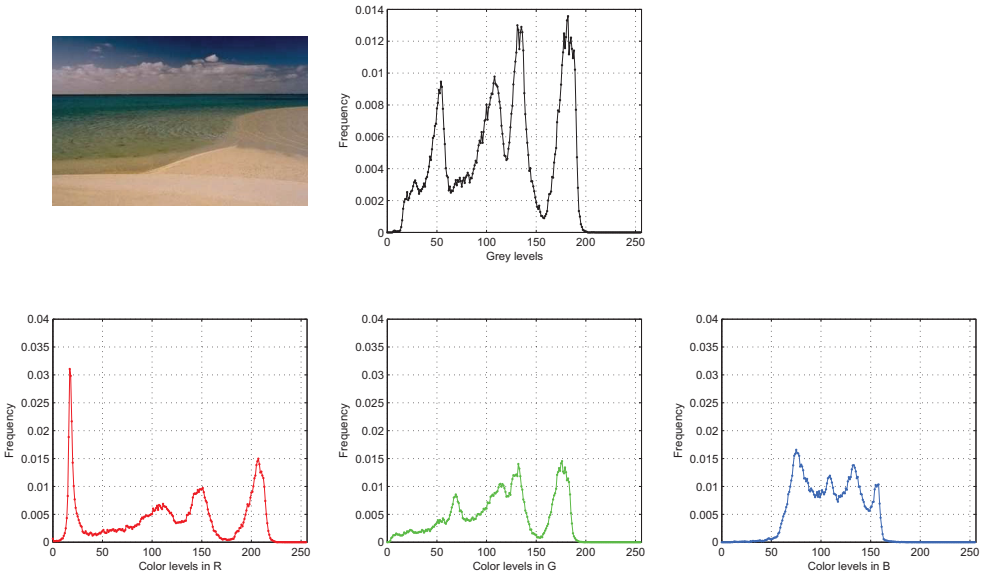

Figure 2.1: An image, its grey level histogram, and the histograms in R, G, and B

The low dimensional color histograms are simple, fast, and have been applied to many applications [63], for example IBM QBIC [25]. Advantages in the color histograms also include rotation invariance, robustness against occlusion and changes of view [70].

However, the color histograms have limitations, and one of the limitations is that no spatial relationship between two colors is included [40, 69, 87, 91]. The correlograms [40] address this problem by incorporating spatial relationships among color levels of neighboring pixels into histograms.

2.1.2 Correlograms and Autocorrelograms

A correlogram [40] of an image describes the joint distribution of two color levels, when the two color levels are spatially at a defined distance [40] in the image. The distance

between two color levels is defined as follows. Consider an image as a two-dimensional matrix with a size as $H \times W$. If one color level c_i is at position $\{x_1, y_1\}, 1 \leq x_1 \leq H, 1 \leq y_1 \leq W$, and the other color level c_j is at position $\{x_2, y_2\}, 1 \leq x_2 \leq H, 1 \leq y_2 \leq W$, the distance between the two color levels c_i and c_j is: $\text{Distance}(c_i, c_j) = \max\{|x_1 - x_2|, |y_1 - y_2|\}$. To extract a correlogram, an image is quantized into n sets of colors $C = \{c_1, \ldots, c_n\}$. A correlogram CH at a distance d is a two-dimensional matrix

$$CH = \begin{pmatrix} ch(1,1) & ch(1,2) & \cdots & ch(1,n) \\ ch(2,1) & ch(2,2) & \cdots & ch(2,n) \\ \vdots & \vdots & \ddots & \vdots \\ ch(n,1) & ch(n,2) & \cdots & ch(n,n) \end{pmatrix} \tag{2.1}$$

with each bin $ch(i, j)$ as the frequency of the existence of the two colors c_i and c_j at the distance d. Thus, a correlogram is indexed in three dimensions: the two color levels, and the distance between the two color levels. The elements in the correlogram are the frequency of the color pairs at a given distance.

The size of a correlogram for an 8-bit grey-level image without any quantization is 256×256. This size is considerably too large to provide efficient image retrieval. The autocorrelograms [40], which only count the frequency of two identical color levels at a given distance, are presented. The autocorrelograms are indexed by two dimensions: a color level, and the distance between the identical two color levels. The elements in the autocorrelograms are the frequencies of the occurrence of the two identical color levels at a given distance.

The discriminating capabilities of the correlograms and autocorrelograms are better than the histogram in many cases, such as an example in Figure 2.2. The example shows two different images have same histograms, but different correlograms and autocorrelograms.

Both the correlograms and the autocorrelograms consider the color levels at the same distance but at different orientations to be the same, according to the definition of the distance function between two color levels. The size of the correlograms and the autocorrelograms would be multiplied if the color pairs at different orientations are considered separately. This simplification minimizes the feature size, but also limits the discrimination capability of the correlograms and autocorrelograms.

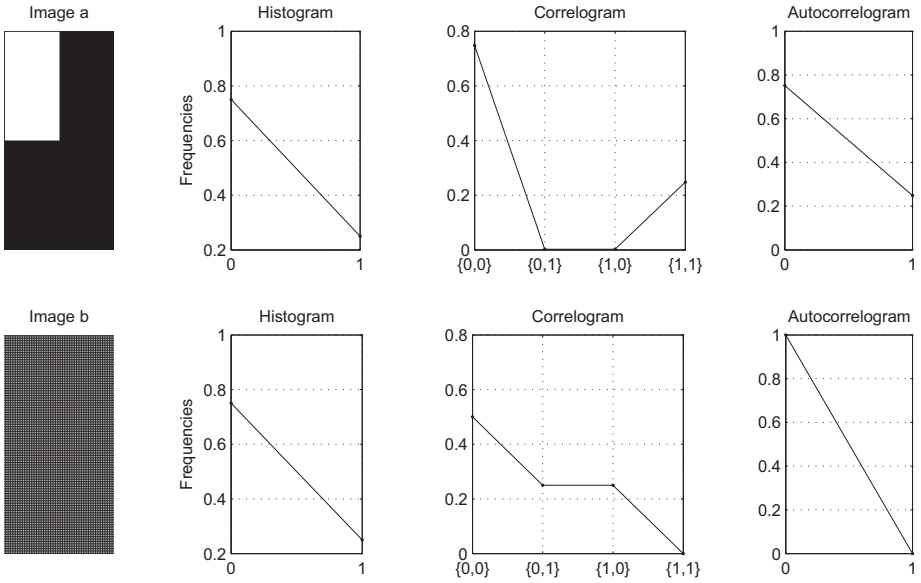

Figure 2.2: Two images a and b with only two color levels 0 and 1. The histograms, the correlograms, and the autocorrelograms are shown from the column 2 to column 4. Both the correlograms and autocorrelograms only show the frequencies of two colors at distance 1.

2.1.3 Local Binary Pattern Histograms

Local Binary Pattern (LBP) [65] is a simple method to amalgamate several orientations while maintaining a small feature size. An LBP level is a number ranging from 0 to 255 which represents the spatial relationships among the 9 color levels in a 3×3 pixel block. The color level of the center pixel is compared with the color levels of its 8 neighboring pixels in the block. The comparison results are transformed into a sequence of binary codes (0,1). When the color level of the neighboring pixel is larger than or equal to the color level of the center pixel, the code is 1. Otherwise, the code is 0. The binary sequence is then converted into an LBP level. A 3×3 block with a center pixel and its neighboring pixels is shown in Figure 2.3(a). The binary code sequence for this block is shown in the Figure 2.3(b).

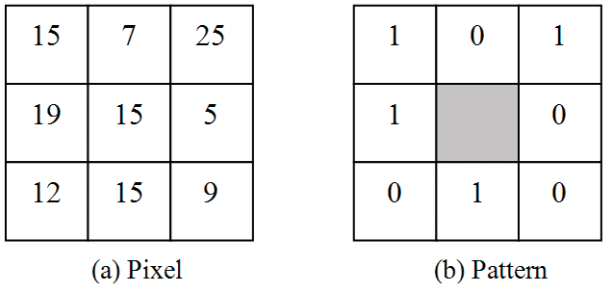

Figure 2.3: A 3×3 pixel block and the corresponding binary sequence in the LBP

Starting from the left top binary code, we consider the eight code sequence in clockwise order as a binary format for an integer. Then the sequence is calculated and transformed to an LBP level as in Equation (2.2).

$$\text{LBP level} = (10100101)_2 = 165 \tag{2.2}$$

Without loss of generality, we assume that the LBP levels are extracted from a grey level image I with the color levels ranging from 0 to 255. Denote the pixel in a 3×3 block as: $c, c_0, \ldots, c_7$ as shown in Figure 2.4 where c is the color level of the center pixel. c_0 to c_7 are the color levels of the eight neighboring pixels. The LBP

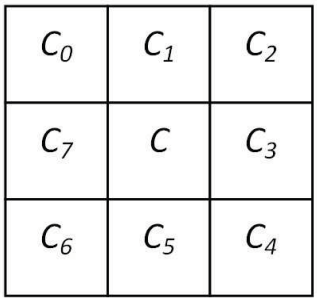

Figure 2.4: Color level notations in a 3×3 pixel block

level of the block is defined as:

$$lbp = \sum_{i=0}^{7} f(c_i - c)2^i \tag{2.3}$$

where the comparison function f is:

$$f(z) = \begin{cases} 1, & z \geq 0; \\ 0, & z < 0. \end{cases} \tag{2.4}$$

Note that the LBP levels are in the range of $\{0, \ldots, 255\}$. Finally the LBP histogram is extracted as $L = \{l_0, \ldots, l_{255}\}$, with each $l_i, 0 \leq i \leq 255$ as the frequency of an LBP level i extracted from all pixels in the image.

2.2 Local Triplet Pattern

2.2.1 Motivations

The LBP maps a 3×3 8-bit two-dimensional integer array from an alphabet of size 256^9 into a single 8-bit integer for an alphabet of size 256. This mapping represents a very high ratio quantization. One of our concerns is whether the quantization alphabet size 256 is the appropriate alphabet size for the quantization.

Moreover, the LBP fails to make distinctions of the patterns generated from two types of blocks. The one type is that the neighboring pixels are with the same color level of the center pixel. The other type is that the neighboring pixels are with larger color levels than the center pixel. These two cases are shown in Figure 2.5. Two

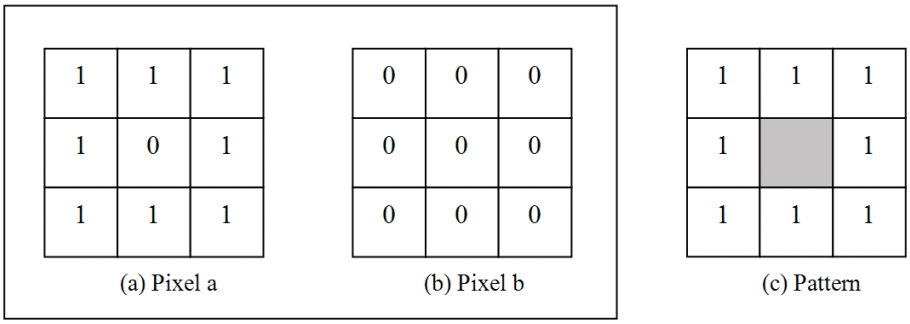

Figure 2.5: Two different 3×3 pixel blocks with a same LBP level

different pixel blocks have a same LBP level although the color levels of the pixels in the blocks are totally different. The comparison function f in Equation (2.4), returns 0 when $z = c_i - c$ is less than 0, and returns 1 when z is larger than or equal to 0. In other words, the resulting binary code for comparing two color levels c_i and c is set as 1 if $c_i \geq c$, and 0 if $c_i < c$. When the binary code "1" is returned, we cannot tell whether c_i is larger than c or equal to c.

Images, especially natural images, have a strong tendency of color continuity, i.e., color levels in a pixel block all have similar colors. This fact indicates that the "equal" condition is not a trivial existence in the color level comparison results. We should consider separating the "equal" condition from the "larger" condition.

These two reasons motivate us to propose a new image feature: Local Triplet Pattern (LTP). The function f is redefined to return three results (smaller, equal, and larger) when comparing two color levels. Each pixel block thus generates a sequence of triplet codes, and the sequence is converted into a finer-grained pattern than the LBP. In addition to the neighboring color changes detected by the LBP, the LTP also detects the continuous color blocks, which brings more discriminating information to image features. In this way, the LTP not only maintains the discriminating capability of the LBP feature, but also is able to distinguish more patterns than the LBP does.

2.2.2 Definition of the LTP

The LTP represents the relationship between two color levels by using three scales, either one is smaller than, equal to, or larger than the other, which can be represented by three numbers (0, 1, 2). The relationships between a color level of a pixel and its eight neighboring color levels are transformed into a sequence of codes, which is converted to an LTP level. A function f' is defined in Equation (2.5) to replace the function f in Equation (2.4) in the LBP.

$$f'(a,b) = \begin{cases} 2, & a > b; \\ 1, & a = b; \\ 0, & a < b. \end{cases} \tag{2.5}$$

As an example shown in Figure 2.6, the newly defined f' function can recognize more different patterns. The three different pixel blocks, which are converted to the same LBP as in Figure 2.5, are converted into different LTPs.

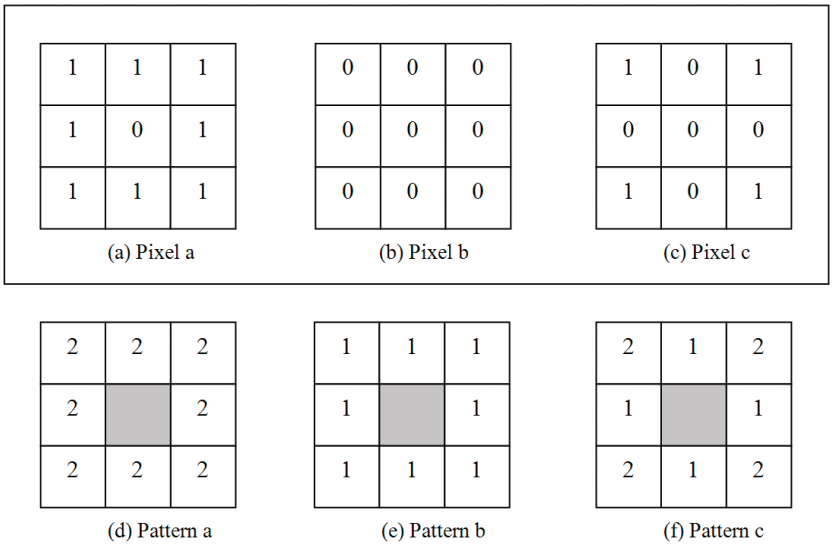

Figure 2.6: Three 3×3 pixel blocks with a same LBP but different LTP

Similar to the LBP encoding, we denote the color level of the pixel in the center as c, and the color levels of the eight surrounding pixels as $c_0, \ldots, c_7$. We assign a factor 3^i to each value of $f'(c_i, c), i = 0, \ldots, 7$, and a 3×3 pixel block is transformed

into a unique pattern level. The LTP is defined in Equation (2.6).

$$ltp = \sum_{i=0}^{7} f'(c_i, c) \times 3^i \tag{2.6}$$

After all the LTPs are calculated from each 3×3 pixel block in an image, an LTP histogram is extracted as the image feature. The LTP histogram is a vector $T = \{t_0, \ldots, t_{6560}\}$, where t_i is the frequency of the blocks with the LTP that are equal to i in the image.

2.2.3 Scaling and Neighboring Parameters

In order to apply the proposed LTP histograms to image retrieval applications, there are two implementation concerns need to be addressed. The function f' in Equation (2.5) has two effects on the LTP histograms. The first is that the LTP histograms are very sparse. The function f' results in 1 if and only if two color levels are the same. Images usually contain color blocks with very similar but not exactly the same color levels. Patterns with 1 (equal) in the triplet codes are always far less frequent than patterns with 0 (less) and 2 (larger). The second consequence is that the LTP feature size (6561) is much larger than the LBP feature size (256).

In order to address these two problems, we introduce a scaling parameter and a neighboring parameter to the feature. An image is scaled from all original color levels into a quantized color space with fewer color levels, which implies all color levels are clustered into several groups. The number of the groups is the scaling parameter S. After the scaling operation, $f'(a, b)$ returns 0 if the color levels a and b belong to the same group, which is very close to how humans perceive the same situation.

The neighboring parameter N is the number of the neighboring pixels which are included to generate an LTP for the block. N is an integer between 1 and 8. In order to cover all the possible combinations between pixel pairs, we suggest that N is at least 4. The size of an LTP histogram with a neighboring parameter N is 3^N. N is a tradeoff between the feature capacity and efficiency. We suggest applying a large neighboring parameter N to provide the best discriminating capability from the LTP features.

Although we only take N neighboring color levels, LTP still captures most joint information of a color level with the other neighboring color levels. For example when

N is 4, the LTP level is calculated from the center pixel c and neighboring pixels c_0 to c_3 as shown in Figure 2.7. The pixels c_4 to c_7 are not included. However, when the c_4 is the center pixel, the previous center pixel c is the neighboring c_0 in this block. Thus, $N \geq 4$ neighboring color levels capture major patterns in an image.

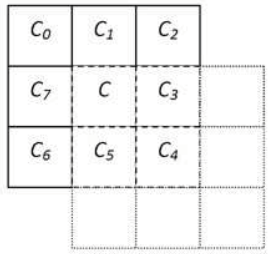

Figure 2.7: Neighboring parameter in the LTP

In summary, by using the scaling parameter, the original color space is quantized to a smaller-sized color space. By using the neighboring parameter, the LTP levels are calculated from a sequence of the triplet codes selected from the 8 neighboring pixels. As a consequence, the LTP histogram becomes much more compact and can be extracted at a flexible length according to the application requirements.

2.2.4 Related Work

Soft LBP Histograms

The soft LBP histograms [1] are an extension from the LBP histograms. In order to make the original LBP more robust to noise, the soft LBP histograms adopt two fuzzy functions to transform the relationship into two numbers, instead of using a binary code to represent the relationship between two color levels of two neighboring pixels. The comparison function f in the LBP is replaced by the two fuzzy functions in Equation (2.7) and (2.8).

$$f_{1,d}(z) = \begin{cases} 0, & z < -d; \\ 0.5 + 0.5\frac{z}{d}, & -d \leq z \leq d; \\ 1, & z > d. \end{cases} \tag{2.7}$$

$$f_{0,d}(z) = 1 - f_{1,d}(z) \tag{2.8}$$

where the parameter d is the control of the fuzziness in the histograms.

The functions $f_{1,d}, f_{0,d}$ from a pixel block in Figure 2.4 contribute a fuzzy weight ranging from 0 to 1 to the bins of the LBP soft histogram. The contribution depends on the binary code of each histogram bin, and the threshold d. Denote the binary code for a soft LBP level s as $\{b_0(s) \dots b_7(s)\}$, where b_0 is the most significant bit. The contribution of the block at position (i, j) to the bin s is then defined as:

$$SLBP_{i,j}(s) = \prod_{k=0}^{7} [b_k(s) f_{1,d}(c_k - c) + (1 - b_k(s)) f_{0,d}(c_k - c)], s = 0, \dots, 255 \quad (2.9)$$

The final LBP soft histograms are the summation of the $SLBP$ for each block in the image.

The fuzzy threshold d in the soft LBP histograms has the similar functionality as the scaling parameter in the LTP histograms. The resulting soft histograms are more robust to noise.

Although the soft LBP histograms have the same alphabet size as of the LBP, they are not as efficient as the LBP. The LBP converts the color level in each 3x3 pixel block into one LBP level. The calculation is simple and fast. The soft LBP histograms link each block in the image with each bin in the pattern histograms. Thus the calculation for the soft LBP histograms increases to the size of the image times the size of the histograms.

Local Ternary Pattern

The Local Ternary Pattern [96] is an image feature very close to our proposed work. For each 3×3 block, the comparison of two color levels also returns three results similar to our proposed LTP histograms. The color levels c of the center pixel is compared with the color levels $c_i, 0 \leq i \leq 7$ of its eight neighboring pixels. The comparison of two color levels returns three results: "smaller" as -1, "equal" as 0, and "larger" as 1. The 3-valued coding includes a threshold t around zero to improve resistance to noise. The comparison is based on three ranges, $[0, c-t], (c-t, c+t), [c+t, 255]$. The "equal" means that the neighboring color level c_i is in the range $c - t < c_i < c + t$. Instead of assigning a 3^n factor to each code, they separate the results into two groups, and calculate two patterns: the upper and the lower patterns. An example of the upper and lower patterns is shown in Figure 2.8. Starting from the upper top binary code

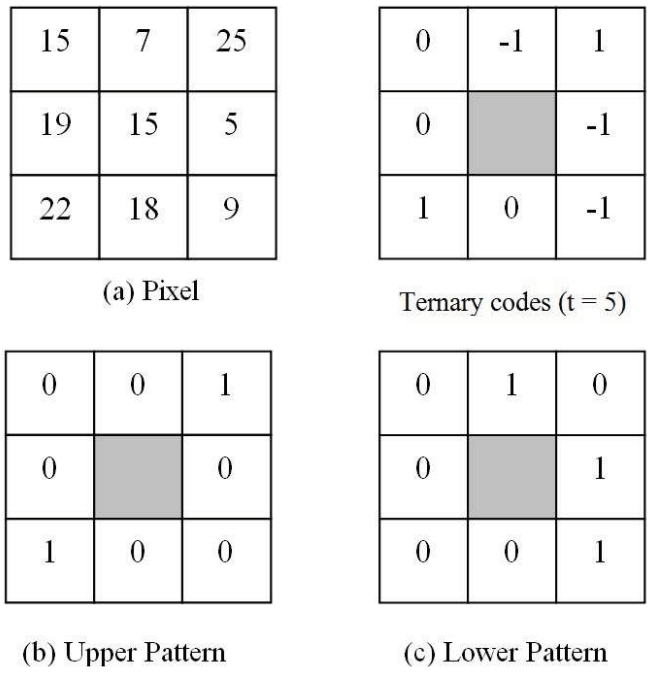

Figure 2.8: Upper and lower patterns in the Local Ternary Pattern

in the clockwise order, the upper ternary codes for this block are 00100010, and the lower ternary codes are 01011000.

Instead of extracting histograms of the patterns, they use a different feature representation and a corresponding distance measure. The distance is given by the following steps. First, the ternary patterns are calculated for each block in the image. Then a binary image b_k is generated for each ternary pattern k, with each value $b_k(i,j)$ in the b_k at position (i,j) is set as 1, if the pattern at position (i,j) is equal to k. Otherwise $b_k(i,j)$ is set as 0. The transform distance matrix d_k is then calculated from the b_k, where the value $d_k(i,j)$ at position (i,j) is the closest distance to the nearby pixels with the same pattern as k. After generating the d_k for both the query X and target Y images, the distance between the two images is defined as:

$$D(X,Y) = \sum_{i \in Y} \sum_{j \in Y} \omega(d^X_{k_Y(i,j)}(i,j), \tau) \tag{2.10}$$

where $d^X_{k_Y(i,j)}$ is the distance matrix from image X, and $k_Y(i,j)$ is the pattern value in the image Y at position (i,j). The function ω is a penalty function which gives penalty based on the value of the closet distance. τ is set as 6 pixels in the reference.

The Ternary pattern features have been applied to face image recognition [96, 21]. The main contribution in the Ternary pattern is the introduction of the transform based distance measure for comparing the face images. The measure is more stable as they capture the same patterns in a close distance area. However, the distance measure also involves much more significant time than the normal measures used for histogram-based features.

The cases of the "larger" and "equal" which are overlapped in the LBP are lightly separated in the Ternary patterns. The Ternary pattern assigns the overlapping cases into two groups, one with the "larger" as in the lower patterns and one with "smaller" as in the upper patterns. However, in each group, the overlapping still exists. Thus, the Ternary patterns make a compromise between the number of the distinguished patterns with the size of the features.

2.3 Experiments

In this section, the performance of the LTP histograms are compared with the related histogram-based features, including the grey-level histograms, the correlograms, the

autocorrelograms, the LBP histograms, soft LBP histograms, and the Local Ternary Pattern histograms. To distinguish our proposed Local Triplet Pattern from the Local Ternary Patterns, LTP is used for our feature, and Ternary is used for the Local Ternary Patterns. Images are first converted into grey level, and all these features are extracted. We use the KL divergence to measure the distance for our LTP histograms. The KL divergence is defined in Equation (1.9) in Section 1.3.3. Their performances are evaluated on the four image data sets, with three from the OUTEX data sets, and the fourth as the Corel1K data set.

2.3.1 Classification of Texture Images

The LBP histograms have been evaluated by their performance of classifying texture images on the following three texture image data sets, Contrib_TC_00006 (TC06), Outex_TC_00013 (TC13) and Outex_TC_00014 (TC14) [73].

The Contrib_TC_00006 data set has 864 texture images in 54 categories, with 432 training images and 432 testing images (8 images per category).

The Outex_TC_00013 data set has 1360 texture images in 68 categories, with 680 training and 680 testing images (10 images per category). The images in these two image data sets are taken under the same illumination condition.

The Outex_TC_00014 data set has 4080 texture images in 68 categories, but with three different illumination sources. The change of the illumination sources puts different shadows on the texture images. This data set shows how the illumination changes affect the performance of the features. 680 images are selected as the training set (10 images per category), and 1360 images are selected as the testing set (20 images per category).

In order to make fair comparisons to the LBP, we apply the LTP histograms to classify images using the same training and testing data. The classifier is the k-NN(k=3) classifier as the same one used by the LBP. For each image in the testing set, the top 3 closest images in the training set are extracted. The class id of the three nearest neighbor training images is assigned to the testing image. The distance between two LBP histograms is measured by the KL divergence in Equation (1.9) in Section 1.3.3. Zeros in the LBP histograms are normalized by a very small value, e.g., 10^{-8} [73].

In Table 2.1, we compare the classification results with the grey level histograms, correlogram, autocorrelogram, LBP histograms, soft LBP histograms, and LTP histograms. The distance measure for the grey level histograms is the KL divergence. The correlogram and autocorrelogram are configured as suggested in [40, 48]. The correlogram features are generated from quantized images with 32 color levels [48]. The autocorrelogram is calculated from quantized images with 64 color levels [40]. The distance measure for the correlograms and autocorrelograms is the L_1 distance measure as suggested in [48].

The threshold d for the soft LBP histograms is selected from 1 to 10, and the best results are list as $d = 2$ for the first two data sets, and $d = 1$ for the third data set. For the Ternary patterns, we extract two histograms, with one from the upper ternary patterns and the other one from the lower Ternary patterns. The threshold t is selected from 1 to 10, and the best results are reported as $t = 2$ for the first two data sets, and $t = 1$ for the second data set. The distance measure for the soft LBP histograms and the Ternary histograms is the KL divergence.

In the LTP histograms, S is the scaling parameter, and N is the neighboring parameter. We show the results from two groups of the LTP histograms with the scaling parameter as 256 and 64, and the neighboring parameter as 8. The features are denoted as LTP S256N8 and LTP S64N8 respectively. The distance measure for The LTP histograms are normalized by 1, and are compared with the KL divergence.

The comparison results show that our LTP histograms outperform the other features on these three texture data sets. The spatial information between the neighboring pixels are included by several features, including the correlograms, the autocorrelograms, the LBP, the soft LBP, the Local Ternary Pattern, and the LTP histograms. The spatial information enables the features to gain better classification results than the grey level histograms. The pattern histograms outperform the correlograms and autocorrelograms because the pattern features recognize the distinction of the neighboring pixels at different orientations. The Ternary histograms achieve better precision than the LBP histograms because they distinguish the pattern more details than the LBP does. Finally, the defined finer-grained patterns in the LTP histograms gain better performance than the other features, as they include the most

Table 2.1: Classification results for the grey level histograms, the correlograms, the autocorrelograms, the LBP histograms, the soft LBP histograms, the Ternary histograms, and the LTP histograms on the TC06, TC13, and TC14 data sets

feature	size	TC06	TC13	TC14
grey level histogram	256	81.0	73.1	37.7
correlogram	1024	93.1	81.0	49.9
autocorrelogram	64	83.3	75.2	42.4
LBP	256	97.7	81.0	60.0
soft LBP ($d = 2$)	256	96.1	84.4	59.6
soft LBP ($d = 1$)	256	96.1	83.7	61.3
Ternary($t = 2$)	512	96.8	84.4	59.9
Ternary($t = 1$)	512	94.4	82.8	60.8
LTP S64N8	6561	**98.6**	**85.0**	57.1
LTP S256N8	6561	97.5	81.8	**64.0**

detailed spatial information into the features.

Because of the illumination changes, the classification error rate of each feature increases on the TC14 data set, especially for the features including the grey level histograms, the correlograms, and the autocorrelograms. The pattern histograms, including the LBP, the soft LBP, the Ternary, and the LTP histograms, are less affected on the TC14 data set. Note that the color levels of pixels in an image will be affected by the illumination condition. Two images with the same content but different illumination conditions have very different grey-level color histograms and correlograms. However, the comparison results of the color levels of neighboring pixels are kept in some areas, because the illumination affects the neighboring pixels in the same manner. Thus, the pattern features which are extracted from the comparison results among color levels are less sensitive to the illumination changes than features that are extracted directly from the color levels.

In Table 2.2, we show the classification results using the LTP histograms with different scaling and neighboring parameters. We mark the numbers in bold when the LTP histograms outperform all the other related features.

From the results in Table 2.2, we observe that the scaling and neighboring parameters affect the LTP histograms in the following aspects. The scaling parameter is the number of the group after the quantization. The images are quantized into continuous color blocks, which provide more occurrences of the neighboring pixel pair

Table 2.2: Classification results for the LTP histograms with different scaling and neighboring parameters

N	size	S	TC06	TC13	TC14
3	27	256	89.4	77.4	48.9
		128	91.7	79.1	47.0
		64	91.7	81.5	45.7
		32	93.8	81.0	42.1
		16	94.7	75.6	40.1
		8	93.1	64.0	31.5
5	243	256	96.3	81.9	54.8
		128	96.3	82.8	54.5
		64	97.5	**84.7**	52.4
		32	97.5	82.6	49.6
		16	97.5	80.7	45.9
		8	95.4	71.2	36.5
8	6561	256	97.5	81.8	**64.0**
		128	**98.1**	82.9	60.0
		64	**98.6**	**85.0**	57.1
		32	**98.6**	**85.0**	53.3
		16	**98.4**	82.1	49.5
		8	97.2	72.2	35.9

Note: We mark the numbers in bold when the LTP histograms outperform all the other related features.

with a same color level. A proper quantization smoothes the image into clustered color blocks, and addresses the sparseness problem in the LTP histograms. However, over-quantization loses too much image detail information, and as a result it becomes difficult to distinguish two different texture images. Thus, the scaling parameter is set as a large number for texture images. The experimental results in Table 2.2 show that the highest precision is achieved when the scaling parameter is 64 for the TC06 and TC13 data sets, and 256 for the TC14 data set.

The neighboring parameter is the number of neighboring pixels that are included in the LTP. Results show that the larger the neighboring parameter, the higher is the retrieval precision. Including more neighboring pixels into the LTP features identifies more patterns, which gives the LTP more discriminate power. Meanwhile, the feature size is increasing from 27 for $N = 3$ to 6561 for $N = 8$. Thus, the neighboring parameter provides a tradeoff between the features' efficiency and effectiveness. When $N = 5$, our LTP histograms gain comparable performance on the TC06 and TC14 data sets, and better performance on the TC13 data set with a smaller size compared to the LBP histograms. When $N = 8$, the LTP histograms outperform the LBP histograms on all these three data sets. For the applications that require fast running speed, $N = 5$ is a good choice for the LTP histograms. Otherwise N is set as 8 to provide the LTP's best capability.

2.3.2 Retrieval on Generic Images

The Corel1K image data set [52] contains 1000 generic JPEG images in 10 classes, with 100 images in each class. To extract features in grey level, all images are first converted into 256 grey level images. Each image serves as a query image. The retrievals are independent of each other. The performance of each feature is measured by the Mean Average Precision (MAP) in Equation (1.14) in Section 1.4.2. The MAP is the average of the precision at which each relevant document is retrieved. Two images are relevant when they are in the same class, and are irrelevant otherwise.

In Table 2.3, we compare the MAP results of the retrievals with the grey level histograms, the correlograms, the autocorrelograms, the LBP histograms, the soft LBP histograms, the Ternary histograms, and the LTP histograms. The setting for the correlograms and autocorrelograms is the same as the features in the Table 2.1.

The threshold values for the LBP soft histograms, and the local Ternary pattern histograms are selected as the best performance one from 1 to 10. The scaling parameter for LTP is 32, and the neighboring parameter is 8. We denote the LTP histograms as LTP S32N8.

Table 2.3: MAP comparison for the grey level histograms, the correlograms, the autocorrelograms, the LBP histograms, the soft LBP histograms, the Ternary histograms, and the LTP histograms

feature	size	MAP
grey level histogram	256	0.305
correlogram	1024	0.371
autocorrelogram	64	0.304
LBP	256	0.437
soft LBP ($d = 4$)	256	0.445
Ternary ($t = 1$)	512	0.434
LTP S32N8	6561	**0.498**

The same results are shown by the MAP comparison as in the classification results in Table 2.1. All the features with spatial information outperform the grey level histograms because that the spatial information improves the discriminating capabilities of the features. The LBP and LTP histograms gain higher MAPs than the correlograms and autocorrelograms because of the different orientation discrepancy in the LBP and LTP. The LTP histograms outperform the LBP histograms and the Ternary histograms because the LTP recognizes more fine-grained patterns.

Table 2.4 shows the MAP results that are achieved by the LTP histograms with different scaling and neighboring parameters. The MAP is marked in bold when the LTP histograms outperform all the other features.

We observe that both the scaling and neighboring parameters have similar effects on the performance of the LTP histograms on both the texture data sets and the generic image data set. The scaling parameter is the number of the quantization color groups before extracting the LTP levels. A proper scaling operation smoothes the image into clustering color blocks, and addresses the sparseness problem in the LTP features. However, if the scaling parameter is too small, the original color space is overly quantized, and images lose too much detail information. As the detail information is less sensitive in the natural images than in the texture images, the

Table 2.4: MAP comparison for the LTP histograms with different scaling and neighboring parameters

N	size	S	MAP	N	size	S	MAP	N	size	S	MAP
3	27	256	0.378	5	243	256	0.409	8	6561	256	0.444
		128	0.376			128	0.415			128	**0.450**
		64	0.373			64	0.425			64	**0.460**
		32	0.404			32	**0.449**			32	**0.498**
		16	0.391			16	0.441			16	**0.483**
		8	0.359			8	0.425			8	**0.464**

scaling parameter is set comparably larger for natural images than for the texture images. A suggested parameter is 16 or 32, for the LTP histograms that are applied on natural images. The neighboring parameter has the same effects as in the experiments on the texture data sets. When $N = 5$, our LTP histograms is smaller sized, but gains higher MAPs than the LBP histograms. In this sense, LTP N5 is a good feature for the applications which have strict requirements on running speed. Without compromising the efficiency, $N = 8$ is suggested to maintain the best discriminating capability from the LTP histograms.

The results on all four data sets show that the LBP and LTP features significantly outperform the other histogram-based features. The performance of the LTP features varies according to different neighboring and scaling parameters. However, the LTP histograms are able to achieve better performance than the LBP histograms when the LTP histograms are with a similar size as of the LBP histograms. When the size of the LTP histograms is larger, the LTP histograms achieve better performance than the LBP histograms with almost every setting of the parameters.

2.4 Summary

In this chapter, we have proposed a new image feature, the Local Triplet Pattern, for image classification and retrieval applications. The LTP is inspired by the LBP feature. An LTP level is generated from a 3×3 pixel block. The color level of the pixel in the center is compared with the color levels of its neighboring pixels. The comparison function in the LBP compares two color levels (a and b) and returns a binary code (0, 1) to indicate two conditions, either $a < b$ or $a \geq b$. The function fails

to distinct the conditions of $a = b$ and $a > b$. We define the comparison function in the LTP to return a triplet code (0, 1, 2) to represent these three conditions and extract a finer-grained pattern. The sequence of triplet codes by comparing with the eight neighboring pixels is converted to an LTP level. Moreover, we include two parameters, a scaling parameter and a neighboring parameter to control the LTP's discriminating capability and efficiency. The scaling parameter addresses the sparseness situation existing in the LTP histograms. The neighboring parameter enables the LTP feature size to be a flexible length according to the specific application requirements. As the LTPs provide spatial relations among the neighboring pixels in a fine-grained level, the LTP histograms achieve better retrieval performance on both the texture image data sets and generic image data set than the related histogram-based features.

Histograms extracted from an image can be either one-dimensional or multi-dimensional. As the example in Figure 2.1, c_i in the one dimensional histograms is either the color levels in each color channel, e.g., the R, G, or B channel, or the grey levels. c_i in the multi-dimensional histograms is a vector with color levels in different channels, such as the vector [R,G,B] in the RGB color space. All image features that we have discussed in this chapter are extracted from grey-level images. The features are small-sized and suitable for fast image retrieval.

Images, however, are usually color images, e.g., in RGB color space. How to effectively utilize the color information in the images is the main problem in Chapter 3. The high-dimensional histograms are very sparse. For example, the three dimensional histogram of a RGB color image is of size 256^3. Suppose the size of the image is 256×256. Lots of bins in the histogram will be zeros or close to zeros. Thus, how to effectively use high-dimensional data is a persisting problem. Quantization on the original color space or model-based solutions are often applied in literature to address the sparsity problem. We will show that the Gaussian Mixture Models (GMMs) are compact and effective representation of the distribution of the high-dimensional data in the images.

Chapter 3

Gaussian Mixture Model-based Image Features

In Chapter 2, we discuss several grey-level histogram-based features, including the grey level histograms, the correlograms, the autocorrelograms, the LBP histograms, the soft LBP histograms, the Ternary histograms, and the proposed LTP histograms. These histogram-based features are extracted from grey-level images and perform highly efficient retrievals. However, the grey level information is usually not sufficient for color image retrieval. Colors, e.g., the RGB colors, can be more important and discriminative than the grey levels. One major difficulty of using the color information lies in that the size of the color features are much larger than the grey-level features. In this chapter, we discuss how to efficiently use the high-dimensional data such as the RGB color information for image retrieval.

One simple way of utilizing the color information in an image is to use multi-dimensional histograms to represent the distributions of the high-dimensional data. However, such high-dimensional histograms are usually of a very large size and inefficient for image matching. For example, the size of a three-dimensional RGB color histogram is 256^3. The Gaussian Mixture Model (GMM) [28, 43, 98, 99] is a very good alternative to the histograms. The GMM is a mixture model with several Gaussian components, which is used to model the empirical distribution of the data. For instance, a GMM can be extracted from the RGB color of all pixels in an image to represent the corresponding RGB color distribution in the image.

How to estimate a GMM from a set of observed data is the main problem we investigate in this chapter. We first review the dominant GMM learning algorithm, the Expectation-Maximization (EM) [15] algorithm. The local maxima problem for the EM algorithm is discussed. When the initial parameters are not selected to be close to the true distribution, the EM algorithm would converge to a local maximum. In order to reduce the dependency on the initial parameter setting and avoid the local

maxima, we then propose the Extended Mass-constraint (EMass) algorithm [38] for estimating the GMMs. The proposed EMass algorithm removes the dependency on the initial parameters, and thus avoids the local maxima. The related deterministic annealing EM (DAEM) algorithm [97] also employs the deterministic annealing method to avoid the local maxima. However, we observe that the DAEM algorithm still has strong dependency on the initial parameters. We test our EMass algorithm and compare it to the EM algorithm and the DAEM algorithm on both simulated data and image data. The results from the simulated experiments show that the EMass algorithm is much more stable than the EM and DAEM algorithm. The EMass always converge to GMMs with high log-likelihood values, while the EM and DAEM algorithms converge to GMMs depending on the initial parameters. From the experiments on the image retrievals, it is shown that the GMMs estimated by the EMass algorithm achieve higher precision than those GMMs learned by the EM and DAEM algorithms.

3.1 Introduction to GMMs

GMMs are compact representations to approximate the data distributions. A GMM is a mixture of several Gaussian components and is determined by a parameter set including: components' weights, means and covariance matrix, which is shown in Equation (3.1),

$$g(y|\Phi) = \sum_{k}^{K} w_k g_k(y|\mu_k, \Sigma_k) \tag{3.1}$$

where y is the observed data. g_k is the kth Gaussian component with $g_k \sim N(\mu_k, \Sigma_k)$, and w_k is the weight of the kth component. To find the GMM for an image is to determine the parameter set $\Phi = \{w_k, \mu_k, \Sigma_k\}$, for $k = 1, \ldots, K$.

The general formula for a one-dimensional Gaussian distribution $N(\mu, \sigma^2)$ is:

$$N(\mu, \sigma^2) \sim \frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{(y-\mu)^2}{2\sigma^2}} \tag{3.2}$$

where y is one-dimensional data, μ is the mean value, and σ^2 is the variance. The mean μ determines the center position of the distribution, and the variance σ^2 determines the shape of the distribution, as an example shown in Figure 3.1.

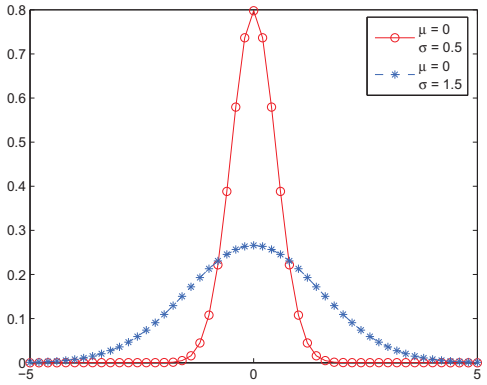

Figure 3.1: One-dimensional Gaussian distribution example

A multi-dimensional Gaussian distribution $N(\mu, \Sigma)$ is defined by

$$N(\mu, \Sigma) \sim \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{(y-\mu)^T\Sigma^{-1}(y-\mu)}{2}\right) \tag{3.3}$$

where y is a d-dimensional data vector, μ is a $d \times 1$ mean vector, and Σ is a $d \times d$ covariance matrix. An example of a two-dimensional Gaussian function (with $d = 2$) is shown in Figure 3.2.

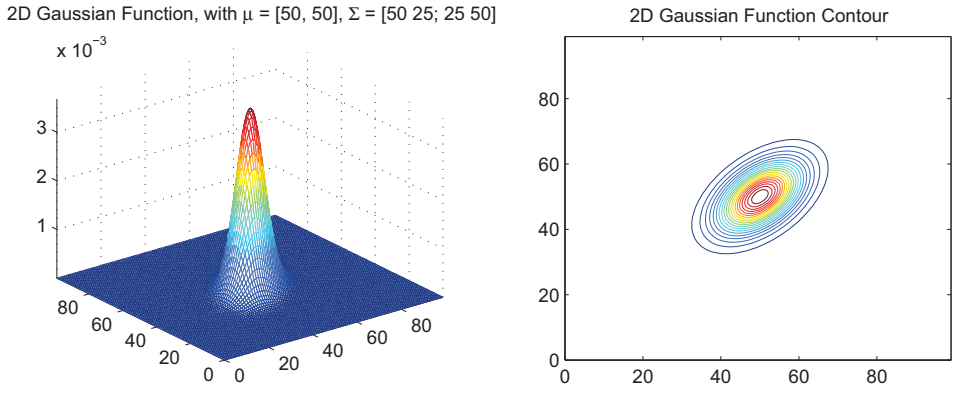

Figure 3.2: Two-Dimensional Gaussian distribution example

We now provide an example to show how GMMs can represent the data distributions in images. Figure 3.3 is an image. Figure 3.4 shows its grey level histogram, and the GMM with 4 components estimated from the grey level data. Figure 3.5 shows its two-dimensional histograms for the B and R colors, and the GMM with

8 components estimated from the color data. The example clearly shows that the GMMs are able to accurately approximate the data distributions.

Figure 3.3: An image of size 256×384

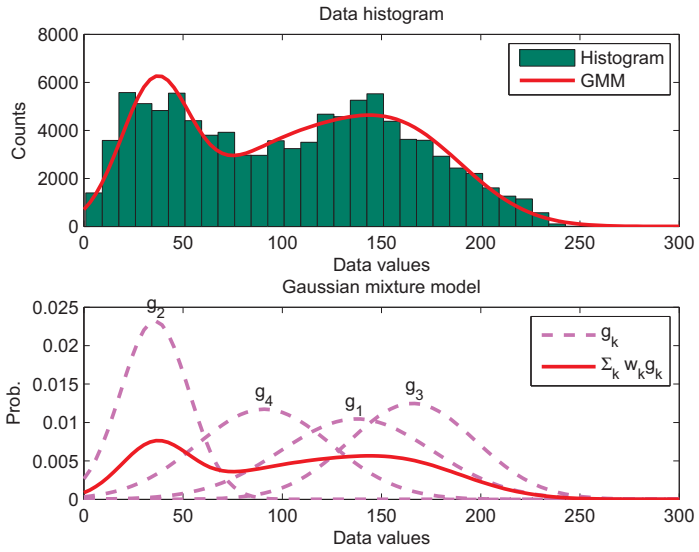

Figure 3.4: One-dimensional histogram (grey level) of the image in Figure 3.3, and its GMM estimation with 4 components

3.2 Estimating GMMs via the EM Algorithm

Without loss of generality, assume that we are looking for a statistical model to represent a group of one-dimensional image data, such as the grey levels. Let Y denote the observed data sequence $Y = \{y_1, \ldots, y_n\}$, where n is the sample size. Given a set of models $\mathcal{F}$ on R^d, we are looking for a model $g(y) \in \mathcal{F}$, which best

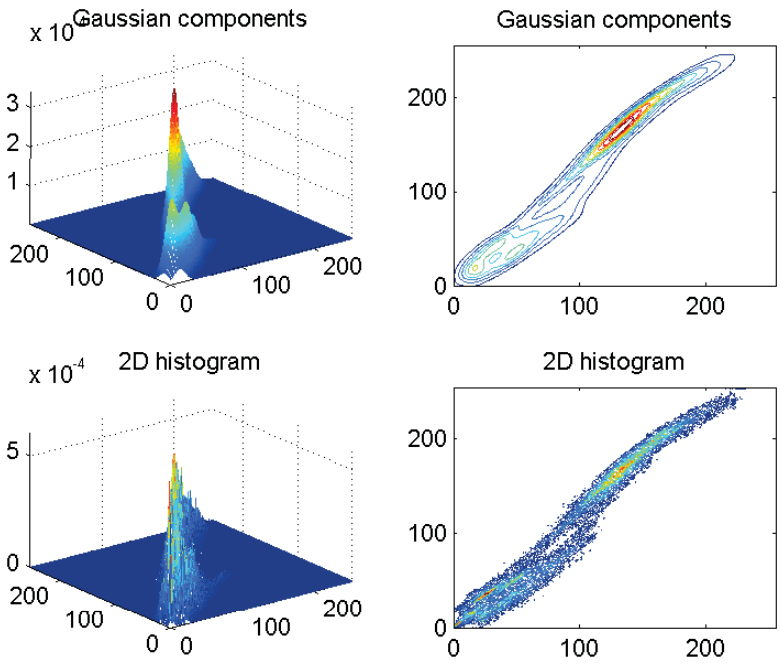

Figure 3.5: Two-dimensional histogram (color levels in BG channels) of the image in Figure 3.3, and its GMM estimation with 8 components

represents the distribution of the given data. If $g(y)$ depends on a set of parameters Φ, $g(y)$ is denoted as $g(y|\Phi)$. One of the popular solutions is to estimate the parameter set to maximize the log-likelihood function. The log-likelihood function $L(\Phi)$ given the observed data Y is defined as

$$L(\Phi) = \log g(y|\Phi) = \log \prod_{i=1}^{n} g(y_i|\Phi) = \sum_{i=1}^{n} \log g(y_i|\Phi) \tag{3.4}$$

where $g(y|\Phi)$ is the model density function that describes the distribution of the observed grey level data Y in the image, and Φ is the parameter set. As we assume that the $g(y|\Phi)$ is a Gaussian distribution, the parameter set Φ consists of two parameters: the mean μ and the variance σ^2. We know that the sample mean and sample variance are the set of parameters which maximizes the log-likelihood function. We can easily calculate the sample statistics, i.e., sample mean, sample variance by Equations (3.5) and (3.6).

$$\mu = E[Y] = \frac{\sum_{i=1}^{n} y_i}{n} \tag{3.5}$$

$$\sigma^2 = E[(Y-\mu)^2] = \frac{\sum_{i=1}^{n} (y_i-\mu)^2}{n} \tag{3.6}$$

where μ is the mean, σ^2 is the variance, and $E[.]$ is the expectation function.

However, if we examine the grey level histogram, such as the example shown in Figure 3.4, we may notice that the histogram is usually not a distribution that can be modeled by only one Gaussian component, but might be modeled by several Gaussian components mixed together. For example, an image with two colors may have two peaks in the histogram, which is modeled better by a GMM with two Gaussian components.

Using GMMs as feature representations has been widely conducted for different applications in many research areas. The Expectation-Maximization (EM) algorithm [15] is the dominant algorithm for estimating GMMs. The EM algorithm estimates a GMM with the highest log-likelihood by two steps: E step and M step. The E step calculates the conditional expectation of the log-likelihood given the observed data. The M step updates the GMM parameters by maximizing the expectation of the log-likelihood.

The log-likelihood function is defined in Equation (3.4). Assume that the $g(y|\Phi)$ is a GMM. A random variable Z is introduced to indicate from which Gaussian

component that an observation y_i of Y is generated. The data is then noted as $X = \{Y, Z\} = \{y_1, z_1, \ldots, y_n, z_n\}$, where $z_i, i = 1, \ldots, n$ is a vector of length K, and K is the total number of components in the GMM. $z_i(k), k = 1, \ldots, K$ is a value between 0 and 1, indicating that y_i is generated by the kth component with a weight $z_i(k)$. Note that the summation of the $z_i(k), k = 1, \ldots, K$ for one observation y_i is 1, i.e., $\sum_{k=1}^{K} z_i(k) = 1$. The complete log-likelihood is defined as:

$$L_c(\Phi) = \log f(y, z|\Phi) = \sum_{i=1}^{n} \log f(y_i, z_i|\Phi) \tag{3.7}$$

where $f(y, z|\Phi)$ is related to the $g(y|\Phi)$ via marginalization $g(y|\Phi) = \int f(y, z|\Phi)dz$.

As we have no information of the hidden variable Z, the log-likelihood $L(\Phi)$ cannot be maximized directly. The EM algorithm obtains a set of parameters Φ that maximizes $L(\Phi)$ by making use of the complete log-likelihood $L_c(\Phi)$ [22]. The parameter set Φ is iteratively updated by maximizing the expectation of the complete log-likelihood $Q(\Phi'|\Phi)$

$$Q(\Phi'|\Phi) = E[\log f(y, z|\Phi')|y; \Phi] = \sum_{y \in \mathcal{Y}} \int_{z \in \mathcal{Z}} k(z|y; \Phi) \log f(y, z|\Phi')dz \tag{3.8}$$

where Φ is the current parameters, Φ' is the parameters to be estimated. $k(z|y; \Phi)$ is the conditional probability function of the hidden variable z given the observed y, and denoted as

$$k(z|y; \Phi) = \frac{f(y, z|\Phi)}{\int_{z \in \mathcal{Z}} f(y, z|\Phi)dz} \tag{3.9}$$

The E step and M step of the EM algorithm are:

E step:

$$Q(\Phi|\Phi^{(t)}) = E[\log f(y, z|\Phi)|y; \Phi^{(t)}] \tag{3.10}$$

M step:

$$\Phi^{(t+1)} = \arg_{\Phi} \max Q(\Phi|\Phi^{(t)}) \tag{3.11}$$

where t is the iteration index.

Dempster et al.[15] indicate that the log-likelihood is non-decreasing in each iteration of the EM algorithm, i.e., each iteration of the EM algorithm is a monotonically increasing process. If the expected complete log-likelihood $L_c(\Phi)$ has several maxima, the EM algorithm converges to either a global or a local maximum depending on the choice of initial value.

3.3 Proposed GMM Estimation Algorithm: Extended Mass-constraint Algorithm

The EM algorithm is an iterative process which updates GMM parameters by maintaining a non-decreasing log-likelihood in each iteration. The EM algorithm does not guarantee to obtain the global maximum, because usually the likelihood function is not strictly concave. Under such cases, the non-decreasing EM algorithm will result in a local maximum or a global maximum depending on the initial parameters. In order to estimate a better maximum, the deterministic annealing algorithm [76, 97] is applied to estimate the GMMs from data.

The deterministic annealing method is first included for data clustering and vector quantization [76]. In this section, we apply the deterministic annealing method to avoid the local maxima problem in the EM algorithm. Our algorithm is motivated by an annealing algorithm for data clustering and vector quantization, which uses the deterministic annealing method and is denoted as the mass-constraint algorithm [76]. The mass-constraint algorithm estimates a group of representative data vectors for clustering data, or a codebook for vector quantization. We extend the mass-constraint algorithm to solve the GMM estimation, which will find the parameter set (weights, mean vectors, and covariance matrix) of a GMM that minimizes the distortion of using the GMM to represent the distribution of the data.

3.3.1 Algorithm Description

We redefine the GMM estimation problem from a group of observed data as minimizing an expected distortion or cost function. The Extended Mass-constraint (EMass) algorithm for GMM estimation is derived as follows. Let $Y = \{y_1, \ldots, y_n\}$ denote observed data vectors, and $g_k, 1 \leq k \leq K$ denote the Gaussian components to be estimated. K is the number of components. The expected distortion of the GMM given the observed data is defined as in Equation (3.12).

$$D(g|y) = \sum_{i=1}^{n}\sum_{k=1}^{K} p(y_i, g_k) d_L(y_i, g_k) = \sum_{i=1}^{n} p(y_i) \sum_{k=1}^{K} p(g_k|y_i) d_L(y_i, g_k) \quad (3.12)$$

where g is the GMM to be estimated. $p(y_i, g_k)$ is the joint probability of an observed data vector y_i and a Gaussian component g_k. The conditional probability $p(g_k|y_i)$ is

the association probability of a component g_k given an observed data y_i. $d_L(y_i, g_k)$ is the distance from a data vector y_i to a Gaussian component g_k.

$$d_L(y_i, g_k) = \frac{1}{2}(d\log(2\pi) + \log|\Sigma_k| + (y_i - \mu_k)'\Sigma_k^{-1}(y_i - \mu_k)) \tag{3.13}$$

where d is the dimensionality of the data vector y_i. μ_k and Σ_k are the parameters for the k-th Gaussian component g_k. Note that the distance d_L is the negative log-likelihood function. Thus to minimize the distortion function D is equal to maximize the log-likelihood in the EM algorithm.

Unlike the EM algorithm which maximizes the log-likelihood without any constraints, we apply constraints on the minimization. To find the unknown joint probability $p(y_i, g_k)$, we choose the one with the maximum entropy. The entropy of the $p(y_i, g_k)$ is

$$H = -\sum_{i=1}^{n}\sum_{k=1}^{K} p(y_i, g_k)\log p(y_i, g_k) \tag{3.14}$$

Minimizing the distortion D and maximizing the entropy H under the constraint that $\sum_{k=1}^{K} p(g_k) = 1$ lead to minimize a Lagrangian

$$F = D - TH \tag{3.15}$$

where T is named as the temperature parameter [76].

Minimize the Distortion

The minimization is performed by two steps. First, we fix the Gaussian components and minimize the Lagrangian F with respect to the association probability $p(g_k|y_i)$. According to the derivation in [76], this step calculates the probability $p(g_k|y_i)$ as

$$p(g_k|y_i) = \frac{p(g_k)\exp\left(-\frac{d_L(y_i,g_k)}{T}\right)}{Z_{y_i}} \tag{3.16}$$

where Z_{y_i} is partition function as:

$$Z_{y_i} = \sum_{k=1}^{K} p(g_k)\exp\left(-\frac{d_L(y_i, g_k)}{T}\right) \tag{3.17}$$

Putting Equation (3.16) into Equation (3.15), we have the minimized Lagrangian F^* with respect to the probability $p(g_k|y_i)$ as:

$$\begin{aligned} F^* &= \min(F) \\ &= -T\sum_{i=1}^{n} p(y_i)\log\sum_{k=1}^{K} p(g_k)\exp\left(-\frac{d_L(y_i,g_k)}{T}\right) \end{aligned} \tag{3.18}$$

Then we fix the association probability $p(g_k|y_i)$ and minimize the Lagrangian F^* with respect to the GMM, which gives

$$\frac{1}{n}\sum_{i=1}^{n} p(g_k|y_i)\frac{d}{dg_k}d_L(y_i,g_k) = 0 \tag{3.19}$$

where $p(g_k|y_i)$ is the association probability in Equation (3.16).

In the mass-constraint algorithm, K. Rose [76] updates the representative vector (μ) for each cluster iteratively. We need to update both the mean and the covariance matrix for all components in GMMs in each iteration. To update the mean μ_k for each component g_k, we put the distance Equation (3.13) into Equation (3.19). The parameters μ_k, Σ_k that minimize F^* satisfy:

$$\sum_{i=1}^{n} p(y_i,g_k)\frac{\partial}{\partial\mu_k}\left(d\log(2\pi)+\log|\Sigma_k|+(y_i-\mu_k)'\Sigma_k^{-1}(y_i-\mu_k)\right)=0$$
$$\sum_{i=1}^{n} p(y_i,g_k)\frac{\partial}{\partial\Sigma_k}\left(d\log(2\pi)+\log|\Sigma_k|+(y_i-\mu_k)'\Sigma_k^{-1}(y_i-\mu_k)\right)=0$$

Thus the mean μ_k and covariance matrix Σ_k are updated as

$$\mu_k = \frac{\sum_{i=1}^{n} y_i p(y_i)p(g_k|y_i)}{p(g_k)} \tag{3.20}$$

$$\Sigma_k = \frac{\sum_{i=1}^{n}(y_i-\mu_k)(y_i-\mu_k)^T p(y_i)p(g_k|y_i)}{p(g_k)} \tag{3.21}$$

where

$$p(g_k|y_i) = \frac{p(g_k)\exp^{(-d_L(y_i,g_k)/T)}}{\sum_{i=1}^{K} p(g_i)\exp^{(-d_L(y_i,g_i)/T)}} \tag{3.22}$$

$$p(g_k) = \sum_{i=1}^{n} p(y_i)p(g_k|y_i) \tag{3.23}$$

In summary, the EMass algorithm starts from a parameter T defined in Equation (3.15), and updates GMMs iteratively by two steps.

Step 1. fix the component parameters, and update the association probabilities by Equation (3.16).

Step 2. fix the association probabilities, and optimize the component parameters by Equations (3.20) and (3.21).

The GMMs are updated by these two steps at a given parameter T until the change of the distortion is smaller than a predefined threshold. We consider the updating process at the T is converged. The parameter T is then decreased to a smaller number, and the GMMs are updated by the two steps again. The iteration keeps updating until the T reaches 0, when the Lagrangian F is equal to D. Thus, the distortion D is minimized in the end.

Phase Transition and the Parameter T

The parameter T starts at a high number, and iteratively lowered at each round until reaching 0. When T is a large number, the association probability of the data and each Gaussian component is uniform. At the early stage, the uniform association probability makes the mixture components evenly cover all the data vectors. The objective function F is minimized mainly by maximizing the entropy. The local maxima due to improper initial data are then avoided. When T is lowered, the association probabilities of a data to different components start to vary according to the distance from the data to the Gaussian components. The objective function F is then minimized by the combination of the distortion and entropy. When the parameter T reaches 0, the distortion is minimized.

The EMass algorithm starts from one Gaussian component at a given T, and increases the number of components at a determined T which is decided by the component covariance matrix. The step to find the critical point to increase the number of Gaussian components is called phase transition check. According to the transition condition in [76], a Gaussian component g_k is split in two components when T is lower than a critical point T_c. The critical point is twice of the largest eigenvalue of the component covariance matrix Σ_k. At the initial stage, we only have one component. The initial parameter T is provided by the initial covariance matrix Σ_1, and set as a value larger than $2\lambda_{max}(\Sigma_1)$.

The Algorithm

The detailed steps of the EMass algorithm for estimating GMMs are given in Algorithm 1. The mean and covariance matrix for the k-th component g_k are updated as

Algorithm 1 Extended Mass-constraint algorithm

Input: data $Y = \{y_1, \ldots, y_n\}$, where n is the number of all data vectors,
number of mixture components K_{max},
minimum temperature T_{min}.

Output: Gaussian components weight $p(g_k)$, mean μ_k, and covariance matrix Σ_k, $k = 1$ to K_{max}.

{Estimate Gaussian mixture model that best describes data $Y = \{y_1, \ldots, y_n\}$}

1: Initialize: $K = 1, \mu_1 = \sum_{i=1}^{n} y_n p(y_i)$, $\Sigma_1 = cov(y)$, $p(g_1) = 1$, and initial $T = 2\lambda_{max}(\Sigma_1)$;
2: **repeat**
3: **repeat**
4: **for** every $k = 1, \ldots, K$ **do**
5: update mean μ_k
6: update covariance matrix Σ_k
7: **end for**
8: perform singularity check on each component
9: **until** converged
10: Cooling step: $T = \alpha T, 0 < \alpha < 1$.
11: **if** $k < K_{max}$ **then**
12: PhaseTransitionCheck
13: **end if**
14: **until** $T \leq T_{min}$

in Equations (3.20) and (3.21).

The phase transition check is listed in Algorithm 2. In the splitting step on line 4, the original component to be split is separated in two components. The weight is split in half for each component. The mean of the new Gaussian component is shifted from the mean of the original component by adding a random offset. The covariance matrix is the same as the matrix of the original component.

Algorithm 2 Phase Transition Check

Input: A GMM with k components $g_i, \ldots, g_k$;
current temperature T;

Output: An updated GMM.

{Check whether components should be separated into two components to be updated}

1: **for** every $i = 1, \ldots, k$ **do**
2: Calculate critical temperature T_c for component g_i
3: **if** $T \leq T_c$ **then**
4: split the component g_i into two components;
5: $k = k + 1$;
6: **end if**
7: **end for**

The singular check on line 8 in Algorithm 1 is to eliminate singular components. Unlike clustering or vector quantization [76], which returns a representative or mean vector for each group, the GMM estimation returns parameters with both the mean vectors, and covariance matrix. In some cases, the covariance matrix encounters singular problems, at which point the algorithm will fail to find the correct GMMs. The singular matrix happens when all elements in a component are too similar to each other, which makes the component narrow in a small gap, and more like a constant. We propose to avoid this problem by singular check in each iteration, maintaining a list with all singular matrix components, removing all singular component related data, and updating the GMM components parameters with the remaining data. All parameters are reset to initial values, and the algorithm goes back to the beginning to train a model based on the remaining data.

3.3.2 Related Work

The deterministic annealing method has been applied to the EM algorithm to avoid local maxima in other works, e.g., the Deterministic Annealing EM algorithm (DAEM). The DAEM algorithm reformats the maximum likelihood estimation into optimizing the expected complete log-likelihood with certain constraints. Let Y denote observed

data vectors, and Z denote the missing data as defined in the EM algorithm. The objective is to maximize the log-likelihood function,

$$Q(\Phi'|\Phi) = E[\log f(y, z; \Phi')|y; \Phi] = \sum_{y \in \mathcal{Y}} \int_{z \in \mathcal{Z}} k(z|y; \Phi) \log f(y, z; \Phi) dz \tag{3.24}$$

and also maximize the entropy

$$H = -\sum_{y \in \mathcal{Y}} \int_{z \in \mathcal{Z}} k(z|y; \Phi) \log k(z|y; \Phi) dz \tag{3.25}$$

under the constraint:

$$\int_{z \in \mathcal{Z}} g(z|y; \Phi) dz = 1 \tag{3.26}$$

The optimization is to maximize the lagrangian

$$F = Q + \frac{1}{\beta} H + \lambda(\int k(z|y; \Phi) dz - 1) \tag{3.27}$$

where β and λ are Lagrange multipliers.

By maximizing the lagrangian with respect to the density function $k(z|y; \Phi)$, we have

$$k(z|y; \Phi) \quad = \quad \frac{1}{Z} \sum_{z} f(y, z|\Phi)^{\beta} \tag{3.28}$$

where Z is the partition function that satisfies

$$Z = \int_{z \in \mathcal{Z}} f(y, z|\Phi)^{\beta} dz$$

The DAEM algorithm is summarized as performing the following E and M steps iteratively at different β, which starts from a small number close to 0, and increases to 1 progressively by multiplying a constant larger than 1, e.g., 1.1.

E step:

$$U(\Phi|\Phi^{(t)}) \quad = \quad E[-\log f(y, z|\Phi)|y; \Phi^{(t)}] + \frac{1}{\beta} E[-\log k(z|y; \Phi)|y; \Phi)] \tag{3.29}$$

M step:

$$\Phi^{(t+1)} = \arg\max_{\Phi} U(\Phi|\Phi^{(t)}) \tag{3.30}$$

where t is the iteration index.

From Equation (3.28), we can see that two special cases of β are 0 and 1. If $\beta = 0$, the posterior probability is a uniform distribution. If $\beta = 1$, the posterior probability is the same as the EM algorithm. When β changes from 0 to 1, the effect of original posterior density on the parameter estimation is increasing as well. The posterior density $k(z|y;\Phi)$ is not reliable at the early training stage, but is getting stable while we are approaching the final solution. Thus the initial β is set as a small number, and increased progressively. The change of β makes the algorithm more stable.

The objective function in Equation (3.27) is the same as the objective function in Equation (3.15) in the EMass algorithm. To minimize the distortion by using the distance function d_L in Equation (3.13) actually maximizes the log-likelihood. Meanwhile, maximizing the joint entropy in Equation (3.14) is also the same to maximize the conditional entropy in Equation (3.25), as the source entropy $H(Y) = -\sum_{i=1}^{n} p(y_i) \log p(y_i)$ in the joint entropy is a constant given the data and can be dropped. The constraint that the component weights must sum to 1 is the same for both algorithms.

The difference between these two algorithms is in the following two aspects: the varying range of the parameter T and β, and the phase transition in the EMass algorithm. In the DAEM algorithm, the parameter β updates from a small number close to 0 and stops at 1. When β is 1, we can see that the final objective function F in the DAEM algorithm is equal to maximize the summation of the log-likelihood and the conditional entropy. In the EMass algorithm, the parameter T starts from an initial large number and decreases to 0 finally. Thus the final objective is to minimize the distortion, which is equal to maximize the log-likelihood only.

Second, the EMass algorithm applies the phase transition to find the best T_c to split a component into two components. Starting from one Gaussian component, the initial parameter T is decided by the component covariance matrix. The number of components is increased at the proper time which also depends on the T and the component covariance matrix. Thus the dependency on the initial data is removed. This is quite important to ensure that two components will not be overlapped because of a too large parameter T. In the DAEM algorithm, no similar scheme is included. How to choose good initial parameter β is not discussed. How to decide the initial β is not a trivial problem, as we will demonstrate in the experiments in Section 3.4.1.

Improper initial β will either return GMMs with all components overlapped to each other, or fail in GMMs with local maxima.

As a summary, the suggested initial parameters for the DAEM algorithm include: the number of Gaussian components K, K groups of Gaussian component parameters (weight, mean, covariance matrix), and the initial β. To estimate the best GMM from the data, the DAEM needs to take several runs starting with a different initial β. The total runs are the number of Gaussian components from 1 to K times the number of the possible choices of the initial β. Compared to this, the EMass algorithm only requires one initial parameter K. Starting from one component, the parameter T is decided by the component covariance matrix. The EMass algorithm updates the number of Gaussian components depending on the Gaussian component's covariance matrix, and stops when the T reaches the minimal value. As the EMass algorithm always starts from the definite parameters, the algorithm always converges to the same results. Thus, the EMass algorithm is more stable and efficient than the DAEM algorithm.

3.3.3 Three Algorithms

The three algorithms, the EM, the DAEM, and the EMass algorithms are introduced for estimating GMMs from a group of data. Their differences lie in that these three algorithms have different initial and final objective functions. The EM algorithm maximizes the expected complete likelihood function to obtain the resulting model parameters. The DAEM algorithm starts from maximizing the conditional entropy, and finalizes by maximizing the summation of the expected log-likelihood function and the entropy. The EMass algorithm starts from maximizing the joint entropy, and finalizes by minimizing the expected distortion function. These differences are listed in Table 3.1.

Table 3.1: Difference among the EM, the DAEM, and the EMass algorithms

algorithm	initial objective	final objective
EM	maximize log-likelihood	maximize log-likelihood
DAEM	maximize conditional entropy	maximize log-likelihood + maximize entropy
EMass	maximize joint entropy	minimize distortion

From this Table we can clearly see that the objective of the EM algorithm is the maximization the expected objective function, while the DAEM and EMass algorithms starts from maximizing entropy and stops at different objectives to avoid local maxima problems. We will compare their performance by the experiments in the following section.

3.4 Experiments

We provide two groups of experimental results in this section. The first group of experiments is to estimate a four component GMM from a set of simulated data. The EM, DAEM, and EMass algorithms are applied to train GMMs from the data, and the log-likelihoods are compared.

The second group of experiments is conducted for the Corel1K image data set [52]. The experiments are configured as in Section 2.3.2. These three algorithms are then applied to train GMMs from RGB color data in images. We compare the MAP results of the retrievals by using the GMMs estimated by these three algorithms.

3.4.1 Experiments on Simulated Data

We compare the performance of the EM, DAEM, and EMass algorithms on a simulation data set. The simulation test data are 1600 two-dimensional data generated by 4 Gaussian components. The parameters of the 4 Gaussian distributions are shown in Table 3.2, where the $\alpha_k, k = 1, \ldots, 4$ are the weights, the $\mu_k, k = 1, \ldots, 4$ are mean, and the $\Sigma_k, k = 1, \ldots, 4$ are covariance matrix. The log-likelihood value of all simulated data with the given 4 Gaussian components is -6063. The initial parameters for the EM, and the DAEM algorithms are also listed in Table 3.2. Four means are chosen first. The weight and covariance matrix are calculated based on the mean values.

The parameters of the GMMs with 4 components estimated by the EM, DAEM, and EMass algorithms are listed in Table 3.3. Figure 3.6 to 3.8 show the corresponding GMMs estimated from the simulation data by these three algorithms respectively. The data are shown as red dots in each Figure. The Gaussian components of GMMs

Table 3.2: True and initial parameter sets in simulation experiments

Para.	True values	Initial	Para.	True values	Initial
α_1	0.1875	0.05	Σ_1	$\begin{pmatrix} 0.5 & 0 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0.1 & -0.03 \\ -0.03 & 1.94 \end{pmatrix}$
α_2	0.3125	0.10			
α_3	0.25	0.55	Σ_2	$\begin{pmatrix} 0.5 & 0 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0.08 & -0.06 \\ -0.06 & 1.61 \end{pmatrix}$
α_4	0.25	0.30			
μ_1	(0 0)'	(-1 0)'	Σ_3	$\begin{pmatrix} 0.5 & 0 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1.6 & 0.93 \\ 0.93 & 1.78 \end{pmatrix}$
μ_2	(0 5)'	(0 0)'			
μ_3	(2 7)'	(2 6)'	Σ_4	$\begin{pmatrix} 0.5 & 0 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1.8 & 0 \\ 0 & 1.03 \end{pmatrix}$
μ_4	(4 0)'	(1 0)'			

are represented by ellipses, with the center as the component mean, and the shape and size set by the component covariance matrix.

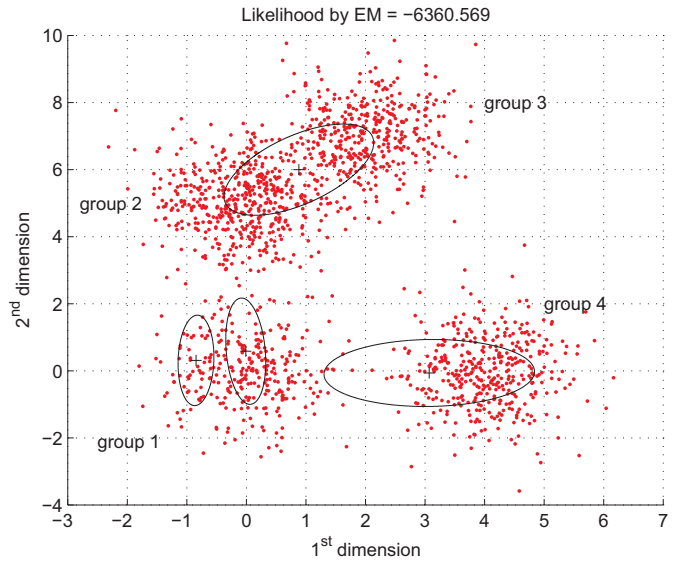

Figure 3.6: Simulation experimental results by the EM algorithm

Figure 3.6 shows the GMM trained by the EM algorithm. Two components of the four components are in the first group of data (group 1), and the other two components cover the remaining three groups of data. Such results are caused by the improper initial data with two starting components with mean vectors as $(-1, 0)', (0, 0)'$. These two components are both closer to the first group of data, and thus the GMM by the EM algorithm limits two components in group 1, and finally converges to the local maximum. This local maxima problem is what we need to solve by applying

Table 3.3: Parameters estimated by the EM, the DAEM, and the EMass algorithms in simulation experiments

Parameters	EM	$DAEM^1$	EMass
α_1	0.033	0.188	0.189
α_2	0.082	0.306	0.307
α_3	0.548	0.256	0.254
α_4	0.337	0.250	0.250
μ_1	(-0.847, 0.309)'	(0.013 -0.008)'	(0.014 0.008)'
μ_2	(-0.012 0.585)'	(-0.093 5.023)'	(-0.105 5.037)'
μ_3	(0.880 6.00)'	(1.986 7.030)'	(2.019 7.051)'
μ_4	(3.071 -0.066)'	(4.025 -0.050)'	(4.026 -0.050)'
Σ_1	$\begin{pmatrix} 0.091 & 0.030 \\ 0.030 & 1.818 \end{pmatrix}$	$\begin{pmatrix} 0.488 & -0.029 \\ -0.029 & 0.977 \end{pmatrix}$	$\begin{pmatrix} 0.487 & 0.026 \\ 0.025 & 1.001 \end{pmatrix}$
Σ_2	$\begin{pmatrix} 0.113 & -0.113 \\ -0.113 & 2.511 \end{pmatrix}$	$\begin{pmatrix} 0.496 & -0.045 \\ -0.045 & 0.924 \end{pmatrix}$	$\begin{pmatrix} 0.464 & -0.062 \\ -0.062 & 0.874 \end{pmatrix}$
Σ_3	$\begin{pmatrix} 1.579 & 1.007 \\ 1.007 & 1.849 \end{pmatrix}$	$\begin{pmatrix} 0.502 & 0.061 \\ 0.061 & 0.995 \end{pmatrix}$	$\begin{pmatrix} 0.447 & 0.017 \\ 0.017 & 0.957 \end{pmatrix}$
Σ_4	$\begin{pmatrix} 3.127 & 0.071 \\ 0.071 & 1.000 \end{pmatrix}$	$\begin{pmatrix} 0.488 & 0.016 \\ 0.016 & 1.054 \end{pmatrix}$	$\begin{pmatrix} 0.484 & 0.015 \\ 0.015 & 1.052 \end{pmatrix}$
Log-likelihood	-6360.6	-6055.8	-6056.7

1: This group of results is the best results with initial parameter $\beta = 0.6$. Note that the initial β starts from a number between 0.1 and 0.9 with an interval 0.1.

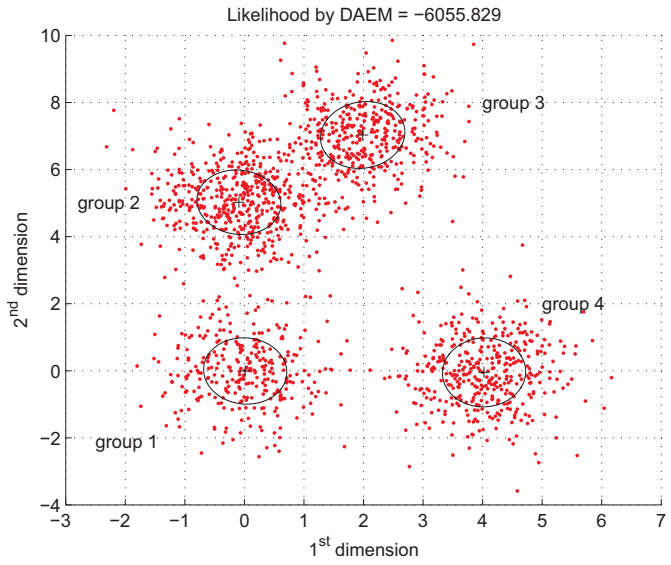

Figure 3.7: Simulation experimental results by the DAEM algorithm

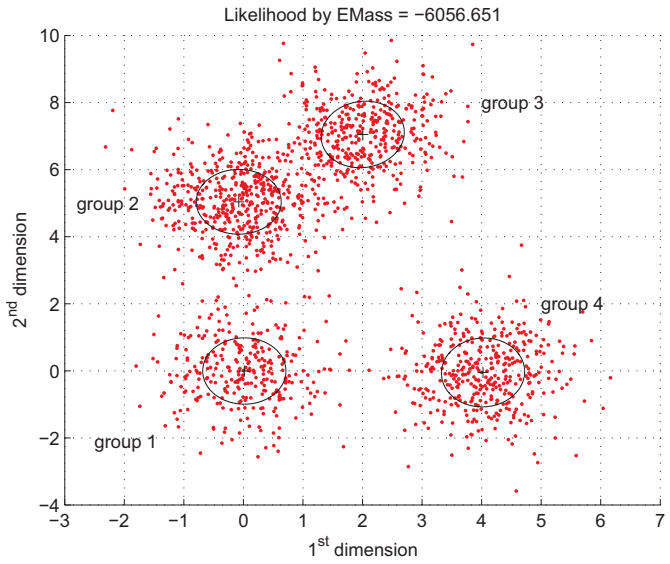

Figure 3.8: Simulation experimental results by the EMass algorithm

the deterministic annealing method. Both the DAEM and the EMass algorithms converge to the GMMs with higher log-likelihoods, which are shown in Figure 3.7 and 3.8. The local maximum caused by the initial data is avoided. The resulting GMMs show four components with each component covering one group of data.

The results in Table 3.3 show that the DAEM algorithm achieves the highest log-likelihood among these three algorithms. However, we also notice that the DAEM algorithm is very sensitive to the choice of the initial parameter β. Figure 3.9 and 3.10 provide two examples that the DAEM algorithm fails to find the correct 4 components with an improper initial parameter β, 0.3 and 0.8. When the β starts from a low value (0.3), the 4 components are overlapped into one component. When the β starts from a high value (0.8), the local maxima cannot be solved by the annealing method. The results indicate that the DAEM has strong dependency on the initial parameter β, which is the main problem we need to solve by applying the annealing method. As no prior information is available to decide the initial value of β, we have to run the DAEM starting from different β and choose the final GMM with the highest log-likelihood. This solution is not efficient.

Figure 3.11 shows the log-likelihood results of 50 runs of each algorithm. The training data is the same 1600 data vectors generated by the parameters in Table 3.2.

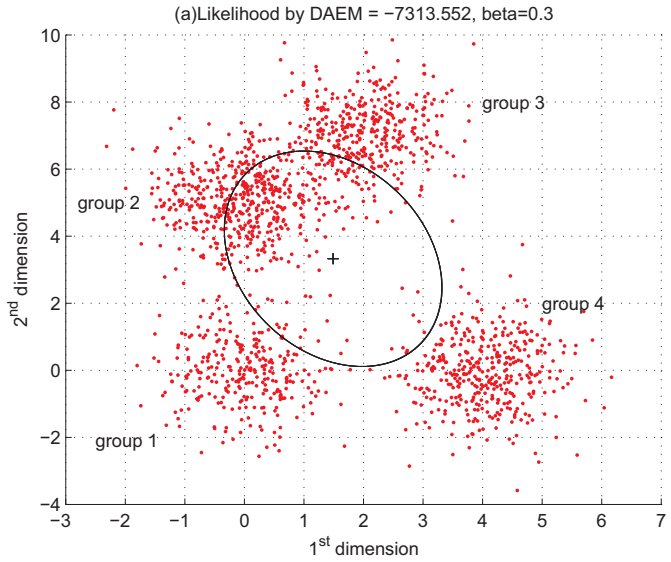

Figure 3.9: Simulation experimental results by the DAEM algorithm. (a): β starts from 0.3.

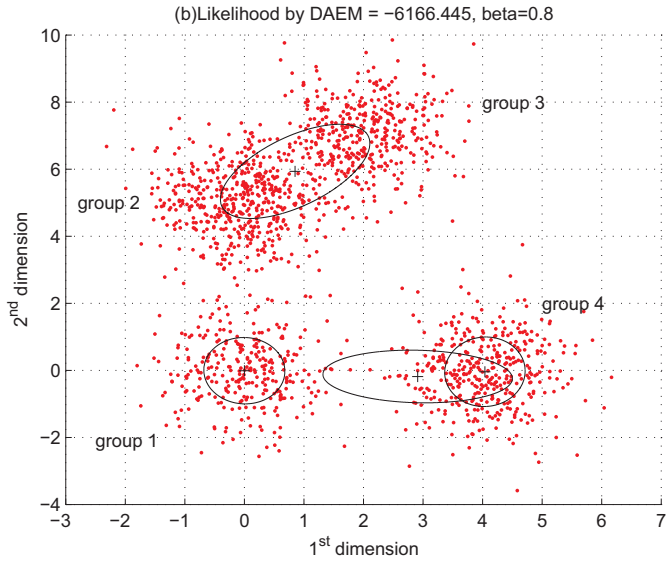

Figure 3.10: Simulation experimental results by the DAEM algorithm. (b): β starts from 0.8.

GMMs are trained by these three algorithms, and the results are shown in 3 bars with each indicating the log-likelihoods of the GMMs estimated by each algorithm for 50 times. The height and value of the bar show the range of the log-likelihoods, and the cross in the bar center is the average log-likelihood of the 50 runs. The initial parameters, e.g., weight, mean and covariance matrix, are randomly generated. The parameter β for the DAEM algorithm is chosen as 0.6 as suggested by the results in Table 3.3. The EM algorithm show strong dependency on the initial parameters. The log-likelihoods of the GMMs fall into a wide range with different values. The results are dependent on the initial parameters. The DAEM algorithm obtains average higher log-likelihoods than the EM algorithm. The average log-likelihood is much closer to the highest log-likelihood. However, in some cases, the DAEM algorithm converges to the GMMs with lower log-likelihoods. These results indicate that the choice of the β and the initial Gaussian components affects the DAEM results. Despite the dependency on the parameter β, the dependency on the initial parameters, e.g., weight, means and covariance matrix, is reduced but not eliminated by applying the annealing method in the DAEM algorithm. Clearly, the EMass algorithm is the most stable algorithm. For all the 50 runs, the EMass converges to the GMMs with the same log-likelihood, which is very close to the true value.

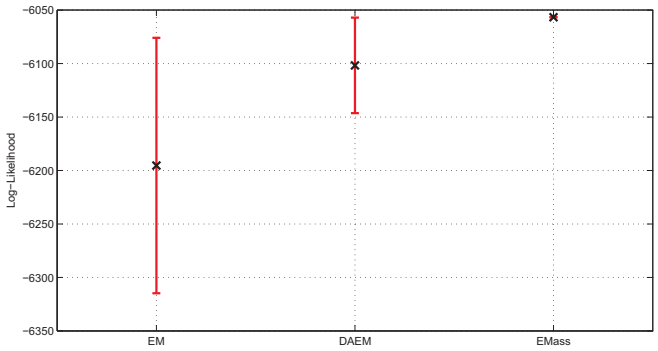

Figure 3.11: The log-likelihood of the GMMs estimated by using the EM, the DAEM, and the EMass algorithms for 50 runs

3.4.2 Retrieval on Generic Images

We evaluate the performance of the image retrieval by using the GMMs estimated from the RGB data in images. The GMMs are estimated by the EM, the DAEM, and the EMass algorithm for the Corel1K image data set [52], which contains 1000 generic JPEG images in 10 classes, with 100 images in each class.

For each image, the RGB color data is represented by a GMM which is estimated by using the EM, DAEM and EMass algorithms. The EM algorithm estimates the GMMs with different number of components from 1 to 16, and selects the GMMs with the highest log-likelihoods. The DAEM algorithm also estimates the GMMs with different number of the components ranging from 1 to 16. In addition, for each number, we run the DAEM with an initial parameter β starting from 0.1 to 0.9 with an interval 0.1. The final GMMs are the ones with the highest log-likelihoods. The EMass algorithm estimates the GMMs with maximum 16 components.

The distance between two GMMs are measured by the unscented transformed (UT) based approximation distance measure [28]. Given two Gaussian Mixture Models $f = \sum_{i=1}^{i} \alpha_i N(\mu_{1,i}, \Sigma_{1,i})$, and $g = \sum_{i=1}^{m} \beta_i N(\mu_{2,i}, \Sigma_{2,i})$. The UT distance measure between f and g is defined as:

$$UT(f, g) \simeq \frac{1}{2d} \sum_{i=1}^{i} \alpha_i \sum_{k=1}^{2d} \log g(x_{i,k}) \tag{3.31}$$

$$x_{i,k} = \mu_{1,i} + (\sqrt{d\Sigma_{1,i}})_k, k = 1, \ldots, d \tag{3.32}$$

$$x_{i,d+k} = \mu_{1,i} - (\sqrt{d\Sigma_{1,i}})_k, k = 1, \ldots, d \tag{3.33}$$

where d is the feature dimensionality, and $d = 3$ for the RGB color space.

The performance is compared to the RGB histograms and the RGB 3D histograms. The RGB histograms are three 256-bin histograms extracted from the R, G, and B channel respectively. The RGB 3D histograms are three-dimensional histograms generated from RGB channels together. The color levels in each channel are uniformly quantized into 8 groups, and the 3D histograms are of the size of $8^3 = 512$. Similarities between two histograms are calculated by the KL divergence in Equation (1.9) in Section 1.3.3. The MAP comparison results are shown in Table 3.4.

From the comparison results in Table 3.4, we observe that the GMMs estimated by the EMass algorithm gain higher retrieval precision than the RGB histograms.

Table 3.4: MAP comparison results with retrievals using different features

	Features	MAP
Histogram-based	RGB histogram	0.398
	RGB 3D	0.447
GMM model-based	EM	0.420
	DAEM	0.433
	EMass	**0.481**

The RGB histograms are three separate histograms in each R,G,B channel. Thus the joint information of the three channels is missing. The RGB 3D histograms lose some information because of the quantization. The quantization is necessary though, because the original histogram size is 256^3. The histogram representation is not suitable for such a large size, while the GMMs can represent the 3D RGB data in its original space.

The GMMs trained by the EMass algorithm achieve higher MAP results than the GMMs trained by the EM and DAEM algorithms. These results are consistent with the results from the simulation experiments. The EMass algorithm is more stable and always achieves GMMs with high log-likelihoods, while the EM and DAEM algorithms converge to the global or local maxima depending on how well the initial parameters are.

3.5 Summary

In this chapter, we present the Extended Mass-constraint (EMass) algorithm to estimate GMMs from data. The EMass algorithm converges to a maximum which has no dependence on the initial parameters. The local maxima problem in the EM algorithm is addressed by the deterministic annealing method in the EMass algorithm. Thus the GMMs estimated by the EMass algorithm are more stable than the EM algorithm. Through the specific choice of the parameter and the phase transition, the EMass algorithm is more stable and efficient than the related DAEM algorithm.

We also show that the estimated GMMs are a good and compact representation for high-dimensional data for image retrieval. We test the performance of the GMMs trained from RGB three-dimensional data. The experimental results show that the GMMs trained by the EMass algorithm achieve much higher precision than the GMMs

trained by the EM algorithm and the DAEM algorithm.

We show that the GMMs are effective representations for high-dimensional data when the histograms are not available. However, the performance of the GMMs depends on how accurate the GMMs can represent the true data distributions. Thus, two problems are intrinsic and difficult to solve. First, the accuracy of how the GMMs can represent the distribution of the data is dependent on the size of data. The GMM estimation algorithm cannot converge to a good result on small sized images. The data in an image with size of 384×256 as in our experiments is still not enough for the 256^3 feature space. Second, we make a strong assumption that the distribution of the multi-dimensional data is a mixture of Gaussian. The method would fail if the data do not have the GMM properties. How to test and evaluate the multi-dimensional data is a GMM distribution is a very difficult problem.

Still working on how to utilize high-dimensional feature data in images, we propose an efficient retrieval method in the next chapter. Motivated by image compression algorithms, we present a retrieval method in the DCT domain based on the hypothesis testing. The method is derived by solid theoretical reasoning, and shows very good performance on image matching.

Chapter 4

JPEG Image Retrieval

In Chapter 3, we show that using the GMMs to model the RGB color information in the images achieves promising performance for image matching. In this chapter, we continue discussing how to effectively utilize the color information in the images. Although the RGB color space is one of the most popular color spaces, more and more images are in different color spaces and in compressed formats, such as the JPEG and JIFF [42], etc. Taking the JPEG standard as an example, the YCbCr color space is adopted instead of the RGB color space. More importantly, as images are in compressed formats, we should consider whether and how to extract image features directly from the compressed domain or partially decompressed data for efficiency.

Following this idea, we first show related work for image matching in the compressed domain. From the review, we notice an important problem in these matching methods in literature. The choices of the features and the distance measurement to compare the features are usually two separate decisions. The design involves a fair amount of heuristic decisions, especially for feature comparison.

In order to reduce the heuristics matter of image matching, we propose an image retrieval scheme (DCT2KL) [39] based on the hypothesis testing. The feature extraction and distance measure are chosen as one whole decision. The DCT2KL algorithm uses the Discrete Cosine Transform (DCT) coefficients restored by partially decoding JPEG images, and derives distance measure accordingly. Given a query image, we form a hypothesis for each target image that its DCT coefficient sequences are emitted from the same sources as the corresponding sequences in the query image. Testing these hypotheses by measuring the log-likelihoods eventually yields a simple yet efficient scheme that ranks each target image according to the Kullback-Leibler (KL) divergence between the empirical distributions of the DCT coefficient sequences in the query image and those in the target image. The detailed steps and derivations are introduced. The performance of the proposed scheme shows that our scheme

achieves consistently better retrieval results than the related methods in literature.

4.1 JPEG Image Standard

Image matching in CBIR consists of indexing and retrieving images based on their content information. A typical image retrieval system first extracts various features, such as color, texture, and shape, from images. The similarity between two images is measured by comparing these features. A persisting challenge in designing these feature-based CBIR systems is to identify and extract effective features for retrieval. Further complicating the design task are the following two facts: first, the number of images to be retrieved is ever rapidly growing, and second, most of these images are stored in compressed format like the popular JPEG. It is thus important to consider image matching by using features extracted directly from compressed or partially decoded images, in particular, the JPEG images.

JPEG(Joint Photographic Expert Group) standard [42] is one of the most popular image formats currently. Most images, whether in an image database or online, are JPEG images because the JPEG standard provides both high compression rate, and good image quality. For example, a 384×256 Bitmap image without compression and its corresponding JPEG file are shown in Figure 4.1. We could hardly tell the visually difference between the two images. However, the Bitmap image is approximately with size $384 \times 256 \times 3 = 294.912$k bytes, and its JPEG file is about 34.7k bytes.

The JPEG standard is a DCT-based coding scheme, which encompasses two components, encoder and decoder. The encoder compresses an uncompressed image into JPEG format, and the decoder does the opposite work.

4.1.1 JPEG Encoder

The JPEG encoder diagram is shown in Figure 4.2. The encoder usually consists of four steps as follows.

Step 1: YCbCr Space and Level Shifting

Uncompressed images are in the RGB color space with the R, G, B channel, with each indicates the color Red, Green, and Blue in the image. The first step to convert RGB

Figure 4.1: An image in Bitmap and JPEG

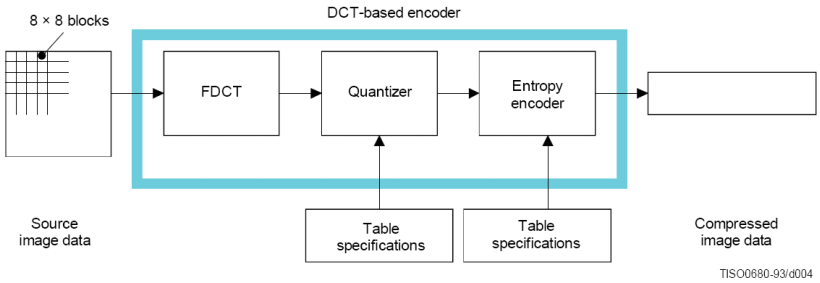

Figure 4.2: JPEG encoder diagram [42]

images to JPEG images is to convert from the RGB space into the YCbCr space.

The conversion from the RGB space to the YCbCr space is:

$$Y = 0.299R + 0.587G + 0.114B \quad (4.1)$$

$$Cb = -0.1687R - 0.3313G + 0.5B + 128 \quad (4.2)$$

$$Cr = 0.5R - 0.4187G - 0.0813B + 128 \quad (4.3)$$

The value Y is called the luminance, which represents the intensity of a color perceived by the human eyes. We observe from the Equation (4.1) that Y actually is a weighted summation of the R, G, and B components. The weights are assigned based on the facts that, human eyes are more sensitive to the Green component, followed by the Red component, and the Blue component at last.

The values Cb and Cr are called the chrominance values and measure the saturation of the color. Cb and Cr indicate how much blue and red exist in the color, respectively.

In the RGB space, we could not tell which channel is more important than the other. However, in the YCbCr space, the Y channel is usually more important than the Cb and Cr channels, as the luminance carries more sensitive data than the blue and red colors. The conversion from the RGB to YCbCr provides a better separation among the three color channels, which allows for different quantization and compression rate for each channel. In this way, JPEG could achieve a higher compression ratio with less quality loss.

In each channel, color levels range from $[0, 2^P - 1]$, where P indicates the maximum bits per pixel is needed in the image. P is 8 for most images. To prepare the data as an input to the next DCT step, which requires the input values to be centered around 0, color levels in YCbCr channels are shifted from the unsigned integers $[0, 2^P - 1]$ to signed integers as: $[-2^{P-1}, 2^{P-1} - 1]$, e.g., [0, 255] to [-128, 127].

step 2: 8x8 FDCT

The image data in the YCbCr is grouped into several non-overlapped 8x8 blocks. Each block is processed by the Forward DCT to obtain 64 DCT coefficients. All blocks are processed from left to right and from top to bottom. The 64 DCT coefficients include 1 DC coefficient and 63 AC coefficients as shown in Table 4.1. The coefficient at the

left and top corner is the DC coefficient, and the 63 AC coefficients are arranged in a zigzag order (AC1- AC63).

Table 4.1: DCT coefficients in ZigZag order

DC	AC1	AC5	AC6	AC14	AC15	AC27	AC28
AC2	AC4	AC7	AC13	AC16	AC26	AC29	AC42
AC3	AC8	AC12	AC17	AC25	AC30	AC41	AC43
AC9	AC11	AC18	AC24	AC31	AC40	AC44	AC53
AC10	AC19	AC23	AC32	AC39	AC45	AC52	AC54
AC20	AC22	AC33	AC38	AC46	AC51	AC55	AC60
AC21	AC34	AC37	AC47	AC50	AC56	AC59	AC61
AC35	AC36	AC48	AC49	AC57	AC58	AC62	AC63

Given the 8x8 source data as $f(x, y), 0 \leq x \leq 7, 0 \leq y \leq 7$. The DCT coefficients $F(u, v), 0 \leq u \leq 7, 0 \leq v \leq 7$ are mathematically defined as:

$$\begin{aligned} F(u,v) &= \frac{1}{4}C(u)C(v)\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y) \\ &\times \cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16} \end{aligned} \tag{4.4}$$

where function $C(\cdot)$ is defined as:

$$C(u) = \begin{cases} \frac{1}{\sqrt{2}}, & \text{if } u = 0; \\ 1, & \text{otherwise.} \end{cases} \tag{4.5}$$

For example, an 8x8 block in the Y channel of an image are given in Table 4.2 After the DCT step by using Equation (4.4), the resulting DCT data are in Table 4.3,

Step 3: Quantizer

In the quantization step, each 8x8 DCT block is quantized by an 8x8 quantization table. Each element in the DCT block is divided by the corresponding value in the quantization table, and rounded to the nearest integer. The quantization table is set either empirically or set by user preference. A typical quantization table is shown in Table 4.4.

The DCT data in Table 4.3 are quantized using Table 4.4, and the resulting quantized DCT data are shown in Table 4.5.

Table 4.2: JPEG encoder example: image data

8	9	10	16	24	40	51	61
67	89	23	11	27	45	22	1
22	38	10	28	38	87	11	12
33	88	29	109	129	11	38	19
3	78	11	28	27	188	17	27
99	23	33	78	12	11	23	11
53	44	123	54	33	231	223	11
14	64	33	53	113	34	66	33
213	11	34	64	22	66	64	22

Table 4.3: JPEG encoder example: DCT data

416.75	23.343	-35.113	118.1	-50.25	31.696	-1.1504	-44.035
-88.654	19.516	-28.426	-29.453	0.25311	-99.395	-25.259	-37.814
-20.404	51.271	55.582	16.167	49.468	8.761	-8.6339	41.63
20.689	-36.748	12.808	-1.1505	-101.71	41.845	14.124	19.465
-54.25	75.377	80.583	25.299	43.75	18.273	52.13	-42.244
9.8952	-35.832	-42.138	-12.549	-37.161	-92.806	-42.42	-5.6387
101	5.7832	6.8661	21.962	-39.974	69.613	-5.5824	22.284
-53.048	87.291	-6.7583	-80.214	141.15	-12.586	-90.133	56.94

Table 4.4: Quantization table example

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Table 4.5: JPEG encoder example: quantized DCT data

26	2	-4	7	-2	1	0	-1
-7	2	-2	-2	0	-2	0	-1
-1	4	3	1	1	0	0	1
1	-2	1	0	-2	0	0	0
-3	3	2	0	1	0	1	-1
0	-1	-1	0	0	-1	0	0
2	0	0	0	0	1	0	0
-1	1	0	-1	1	0	-1	1

Table 4.5 shows an example in the Y channel. As an image has the Y, Cb, and Cr channels, two quantization tables are usually employed with one for Y channel, and one for the Cb and Cr channels. Both quantization tables are saved together with the image data in the JPEG file. During the decoding, these tables are extracted to dequantize the data in order to restore the image data.

Notice that after the quantization, the DCT coefficients in the high index positions are equal or close to zeros. This is an advantage to gain better compression rate in the next step, the entropy encoder.

Step 4: Entropy Encoder

The data after quantization are encoded by the entropy encoder in this step. The JPEG standard suggests two entropy decoders, the Huffman encoder, and the arithmetic encoder. The arithmetic encoded JPEG files are typically 5% smaller than the files encoded by the Huffman encoder. However, the arithmetic coding is not popularly used because of the following two reasons. The arithmetic coders are usually owned by different groups or companies in patent, and they run slower than the Huffman coding. Thus, the Huffman encoder is used in most JPEG images.

The entropy encoding is also processed block by block from left to right and from top to bottom. The 64 DCT coefficients, one DC coefficient and 63 AC coefficients, in an 8x8 block are encoded by different methods.

The DC coefficients are not encoded directly. If we consider all the blocks in an image as a sequence, each block (except the first one) has a previous block. We can calculate the difference of the DC coefficient in the current block and the DC

coefficient in the previous block as Diff $= DC_i - DC_{i-1}$, where i is the index of the current DCT block. For the first DCT block ($i = 1$), the previous DC coefficient is considered as 0. The value Diff is encoded instead of the DC coefficient itself.

The AC coefficients in each block are encoded by the Zero-Run-Length (ZRL) coding. The ZRL coding represents each value except 0 by two numbers: the numbers of zeros preceding the value, and the value.

For example, to encode the quantized DCT block in Table 4.5, we first assume the DC coefficient in the previous block is 20. The block will be represented as: 6, (0,2),(0,-7), (0,-1), (0,2), (0,-4), (0,7), (0,-2), (0,4), (0,1), (0,-3), (0,-2), (0,3), (0,-2), (0,-2), (0,1), (1,1), (0,1), (0,1), (0,3), (1,2), (0, -1), (0,2), (1,1), (0,-2), (1,-1), (2,-2), (1,-1), (1,-1) (0,1), (2,1), (2,-1), (0,1), (5,-1), (1,-1), (0,1), (1,-1), (1,1), (0,1), (4,-1), (0,1), EOB(End-Of-Block). Most AC coefficients at high index positions are zeros and the Run-length encoding succinctly represents all the data in the block.

Normally, four Huffman tables are included for DC or AC coefficients, with two for the DC coefficients, and two for the AC coefficients. In each group of these two tables, one is for the coefficients in the Y channel, and the other for the coefficients in the Cb and Cr channels. All the Huffman tables are also stored in JPEG file header as the quantization tables as well. The output from this step is the compressed image data.

4.1.2 JPEG Decoder

The JPEG decoder diagram is shown in Figure 4.3. Decoding is a reverse process of the encoding.

The compressed image data are input into entropy decoder, dequantizer, and inverse DCT to reconstruct the source image. Before the decoding process, the Huffman tables are extracted from the JPEG file header for entropy decoder, and the quantization tables are extracted for the dequantizer. Then the inverse DCT is applied to the DCT coefficients to restore the RGB colors. The (IDCT) is defined as

$$\begin{aligned} f(x,y) &= \frac{1}{4}\sum_{u=0}^{7}\sum_{v=0}^{7} C(u)C(v)F(u,v) \\ &\times \cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16}, 0 \le x,u \le 7, 0 \le y,v \le 7 \end{aligned} \tag{4.6}$$

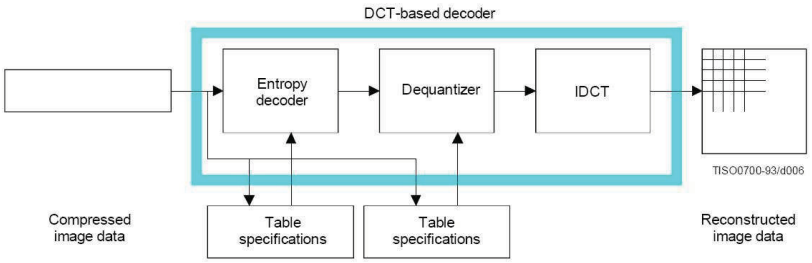

Figure 4.3: JPEG decoder diagram [42]

where $c(\cdot)$ is defined in Equation (4.5).

Given the JPEG image in Figure 4.1, we show how to decode the image to obtain the DCT coefficients step by step. The first step is to retrieve information from the JPEG header, which contains the parameters and tables to decode the image. The tables are the Huffman tables and the quantization tables. The extracted parameters include the image height and width, the index indicating which table should be applied during the decoding. Several important parameters extracted are listed in Table 4.6.

Table 4.6: Decoded image parameters

parameters	value	description
X	384	image height
Y	256	image width
Nf	3	number of the components
Tq	[1,2,2]	quantization table index for each component
Td	[1,2,2]	Huffman table index for the DC coefficients in each component
Ta	[1,2,2]	Huffman table index for the AC coefficients in each component

The quantization tables and Huffman tables are extracted from the JPEG header as well. The header indicates that we have two quantization tables, which are shown in Table 4.7. According to the parameter Nf, we know that the JPEG file has three components, which are usually Y, Cb, and Cr. The parameter Tq = [1,2,2] indicates that the first quantization table is applied to the Y component, and the second quantization table is applied to the Cb, and Cr components.

Table 4.7: Decoded quantization tables

3	2	2	3	5	8	10	12	3	4	5	9	20	20	20	20
2	2	3	4	5	12	12	11	4	4	5	13	20	20	20	20
3	3	3	5	8	11	14	11	5	5	11	20	20	20	20	20
3	3	4	6	10	17	16	12	9	13	20	20	20	20	20	20
4	4	7	11	14	22	21	15	20	20	20	20	20	20	20	20
5	7	11	13	16	21	23	18	20	20	20	20	20	20	20	20
10	13	16	17	21	24	24	20	20	20	20	20	20	20	20	20
14	18	19	20	22	20	21	20	20	20	20	20	20	20	20	20

The extracted Huffman tables for the DC coefficients are shown in Table 4.8. The parameter Td = [1,2,2] indicates that the first Huffman table is applied to the

Table 4.8: Decoded Huffman tables for DC coefficients

Category	Code length	Codeword	Category	Code length	Codeword
0	2	00	0	2	00
1	3	010	1	2	01
2	3	011	2	2	10
3	3	100	3	3	110
4	3	101	4	4	1110
5	3	110	5	5	11110
6	4	1110	6	6	111110
7	5	11110	7	7	1111110
8	6	111110	8	8	11111110
9	7	1111110	9	9	111111110
10	8	1111110	10	10	1111111110
11	9	11111110	11	11	11111111110

DC coefficients in the Y component, and the second Huffman table is applied to the DC coefficients in the Cb and Cr components. The parameter Ta indicates that this JPEG image has two Huffman tables for the AC coefficients. The value Ta = [1,2,2] indicates that the first Huffman table is applied to the AC coefficients in the Y component, and the second Huffman table is applied to the AC coefficients in the Cb, and Cr components.

With all the information extracted from the header, the second step is entropy

decoding, from which we obtain the quantized DCT coefficients from the JPEG encoded data. The DCT blocks are sequentially restored using the corresponding Huffman tables. The quantization tables are applied to dequantize the coefficients to their original value. We can see that the whole process is the reverse of the encoding.

We have introduced the basic steps in the JPEG encoding and decoding processes for JPEG images. Obviously, the original image is not able to be fully reconstructed, because DCT coefficients cannot be fully restored by the dequantization. Thus, the JPEG standard is a lossy compression standard. More details are available in the JPEG standard [42].

In summary, the JPEG standard is a DCT-based image compression scheme. To encode an uncompressed RGB image into a JPEG image, the following four steps are applied: 1) color space conversion from RGB to YCbCr; 2) 8×8 DCT; 3) quantization; and 4) entropy encoding. In order to decode a JPEG image, the above four steps are performed in the reverse order. To obtain the DCT coefficients of a JPEG image, however, we only need to perform two-step partial decoding, without employing the complete decoding process. The images in our data sets, as well as most images that are obtained from online retrieval, are in the JPEG format, so we could extract their DCT coefficients very quickly and easily.

4.2 DCT Domain Image Retrieval by Hypothesis Testing

4.2.1 Motivations and Assumptions

For image retrieval applications, the design of the features and more importantly the associated distance measures for image matching, are often two completely separate decisions. Thus, the design involves a fair amount of heuristic decisions. In an attempt to reduce the dependence on heuristics, in this section, we formulate an image retrieval process as a hypothesis testing problem. Given a query image, we form a hypothesis for each target image that its DCT coefficient sequences are emitted from the same sources as the corresponding sequences in the query image. Testing these hypotheses by measuring the log-likelihoods eventually yields a simple yet efficient scheme that ranks each target image according to the Kullback-Leibler (KL) divergence between the empirical distributions of the DCT coefficient sequences in the query image and

that in the target image.

Our proposed image retrieval scheme works directly on the DCT coefficient sequences restored from JPEG images by performing the first two steps of decoding. In order to further reduce the correlations among DC coefficients in neighboring blocks, we extract all the DC coefficients to construct a sub-image. A 2×2 DCT is then performed on the sub-image. After this process, we scan the blocks in raster order to obtain 67×3 DCT coefficient sequences which consist of 4 sequences from the DC coefficients and 63 AC coefficient sequences in each Y, Cb, Cr channel. These DCT coefficient sequences are indexed in zigzag order [42] except for the first 4 sequences from DC. The process is shown in Figure 4.4.

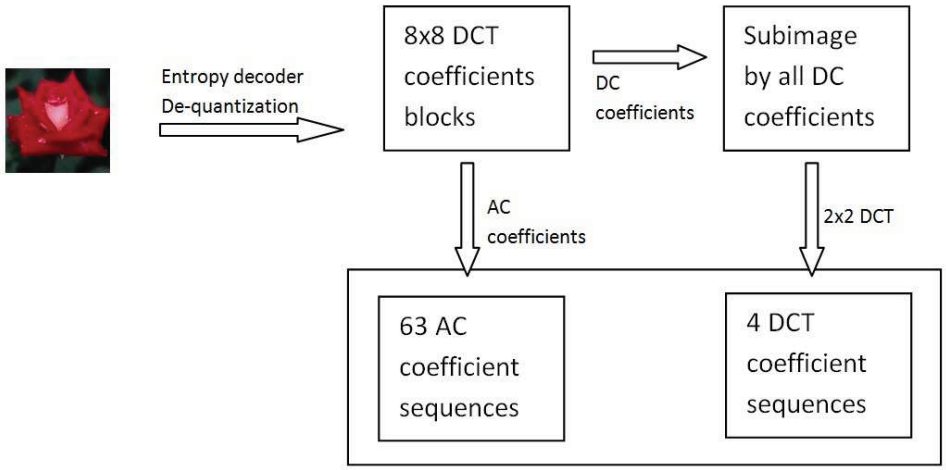

Figure 4.4: Extracting the DCT coefficient sequences from a JPEG image

According to the DCT coefficient decorrelation property, we make assumptions that each sequence is emitted from a memoryless source, and all these sources are independent of each other. The significance of this assumption lies in the following result: it follows from Birkhoff's ergodic theorem [31] that the empirical distribution of an observed sequence is close to the true distribution of the underlying source if the image is large enough. More specifically, we make the following assumptions.

- Each AC sequence is assumed to be emitted from a memoryless source;

- For the DC coefficients, we apply a 2×2 DCT to the sub-image constructed from them, and assume that each of the resulting 4 coefficient sequences is emitted from a memoryless source;

- Due to the well known decorrelation property of DCT, all these sources are assumed to be independent of each other.

4.2.2 Formulation of Image Retrieval as a Hypothesis Testing Problem

Given a query image and N target images, the image retrieval task is to rank the N target images according to their relevance to the query image. For each target image, we form a hypothesis that the DCT coefficient sequences of the query image are emitted from the same sources as the corresponding sequences in the query image. Finding the most relevant target image is thus converted to a hypothesis testing problem. Denote the query image as Q, and the hypothesis corresponding to target image T_i as $H_i, i = 1, \ldots, N$. The probability of the query image Q given the hypothesis H_i is derived as follows.

The DCT coefficients for each channel are processed separately in the same manner. Let us take the luma Y channel as an example. For the DCT coefficients at position $k = 1, \ldots, K$, let $Q_k = \{x_1, \ldots, x_{n_k}\}$ denote the observed kth DCT coefficient sequence for the query image, where n_k is the sequence length, and K is the number of coefficients used. $K = 67$ if all the DCT coefficients are used. The probability mass function (pmf) for the kth DCT coefficient in query image Q is estimated from the empirical distribution of Q_k, denoted as $f_k = (p_1, \ldots, p_{M_k})$, where M_k is the corresponding alphabet size. Similarly, we have the pmf for the DCT coefficients at position k in target image T_i is $g_{ik} = (q_{i1}, \ldots, q_{iM_k})$. Due to the assumed independence among the DCT coefficients at different positions, the probability of the query Q given the hypothesis H_i is calculated as follows.

$$Pr\{Q|H_i\} \;=\; \prod_{k=1}^{K}\prod_{j=1}^{M_k} q_{ij}^{n_k p_j},$$

We also assume that all the hypotheses have the same a priori probability. The retrieval task is then converted to find the most probable hypothesis H_{i^*} among all

the candidates, where i^* is shown as follows

$$\begin{aligned}
i^* &= \arg\max_{1\le i\le N}[Pr\{Q|H_i\}] \\
&= \arg\max_{1\le i\le N}[\log Pr\{Q|H_i\}] \\
&= \arg\max_{1\le i\le N}\log\prod_{k=1}^{K}\prod_{j=1}^{M_k} q_{ij}^{n_k p_j} \\
&= \arg\max_{1\le i\le N}\sum_{k=1}^{K}\sum_{j=1}^{M_k} n_k p_j \log q_{ij} \\
&= \arg\min_{1\le i\le N}\left(\sum_{k=1}^{K} n_k \sum_{j=1}^{M_k} p_j \log p_j - \sum_{k=1}^{K} n_k \sum_{j=1}^{M_k} p_j \log q_{ij}\right) \\
&= \arg\min_{1\le i\le N}\sum_{k=1}^{K} n_k \sum_{j=1}^{M_k} p_j \log \frac{p_j}{q_{ij}} \\
&= \arg\min_{1\le i\le N}\sum_{k=1}^{K} n_k D(f_k||g_{ik}) \qquad (4.7)
\end{aligned}$$

where $D(\cdot||\cdot)$ denotes the KL divergence which is defined in Equation (1.9) in Section 1.3.3. The part $\sum_{k=1}^{K} n_k \sum_{j=1}^{M_k} p_j \log p_j$ can be inserted because it is a constant given a query Q.

Equation (4.7) suggests that the most relevant target image is the one with the least KL divergence to the query image. So we define the distance measure for our proposed retrieval algorithm as the KL divergence between the empirical distributions of DCT coefficients of two images. This solution can be easily extended to retrieve multiple relevant images by ranking all the target images by their KL divergence to the query image.

4.2.3 DCT2KL Algorithm

Our proposed retrieval scheme is summarized in Algorithm 3. It is well known that the DCT coefficients have energy compaction property, which indicates that most energy is compactly saved in the coefficients in the low index positions in zigzag order. Especially the DCT coefficients restored from JPEG images, the DCT coefficients at high positions are zeros because of the quantization step, the information in these coefficients contain less important information for image comparison. In many cases,

Algorithm 3 DCT2KL Image Retrieval Algorithm

Input: A query image Q, target images T_1,..., T_N

Output: A sorted list of all target images

1: Extract K DCT coefficient sequences from Q. Estimate the pmf f_k for each coefficient sequence.
2: **for** each target image T_i **do**
3: Extract K DCT coefficient sequences from each target image T_i. Estimate the pmf g_{ik} for each sequence.
4: Calculate $S(Q, T_i) = \sum_{k=1}^{K} D(f_k||g_{ik})$
5: **end for**
6: Rank all the target images according to $S(Q, T_i)$.

these high indexed DCT coefficients provide no discriminative information among different images. An example in Figure 4.5 is 64 histograms of DCT coefficients restored from the JPEG image in Figure 4.1. These coefficient histograms clearly show the compaction property. Most coefficient histograms at high positions are peaked at zero indicating that coefficients are all zeros. Thus, we suggest a small number of K between 8 and 15 to be used in the DCT2KL algorithm.

Note that the above retrieval scheme can be readily extended to process all types of image formats, by employing additional steps to obtain the DCT coefficients.

4.2.4 Related Work

Feature extraction from JPEG images, or more specifically, their DCT coefficients, for image matching has recently attracted significant interest (see [23, 49, 57, 107, 108] and the references therein). DCT or wavelet coefficients are first reconstructed from compressed images. Features are extracted from these coefficients by different methods.

Some works focus on constructing advanced color, texture or pattern features from DCT coefficients. Various metrics have been applied to measure the distance between two images features. Lay et al. [49] use the energy histograms of the low frequency DCT coefficients as image features. The DCT coefficients are selected from 6 blocks. As the low frequency DCT coefficients carry the most of the energy in DCT blocks,

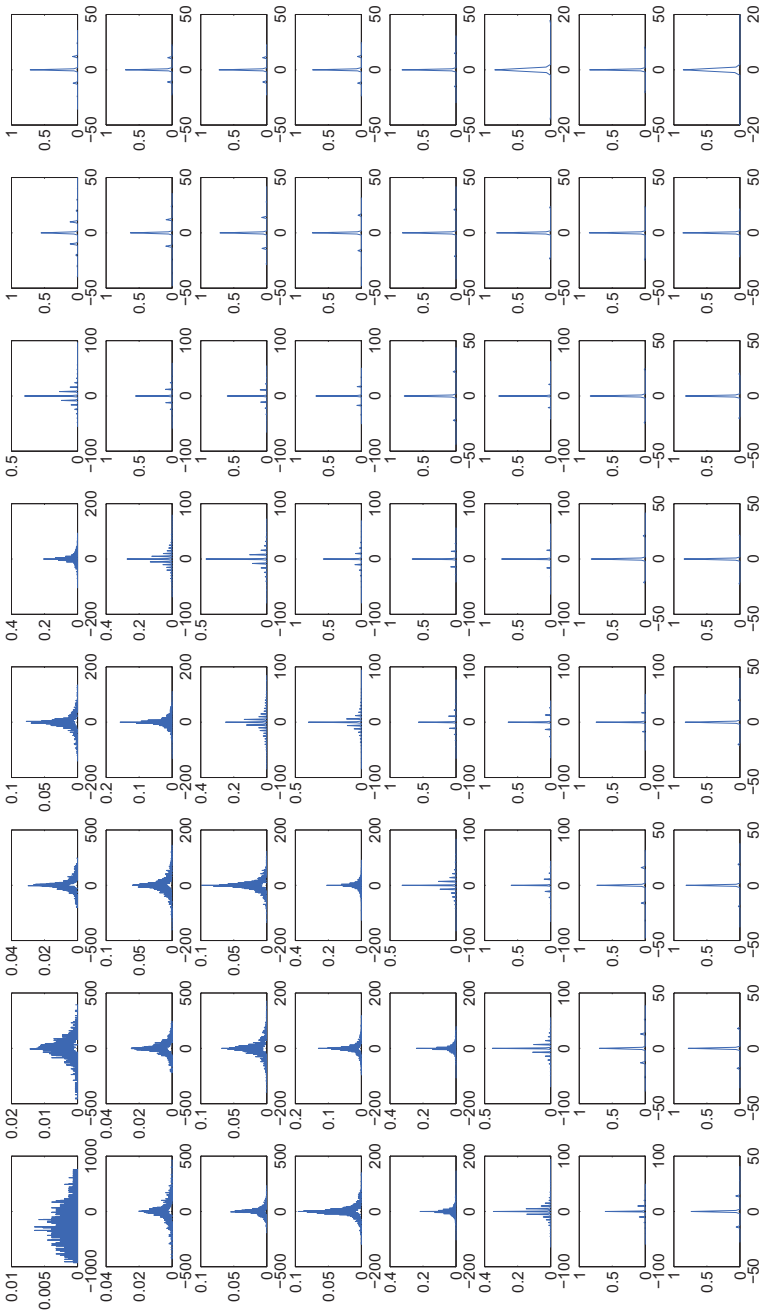

Figure 4.5: DCT coefficient histograms from the JPEG image in Figure 4.1

the 6 blocks contain different combinations of the DCT coefficients in low-frequency are selected. Six DCT coefficient histograms extracted from these blocks are the image features. A normalized Minkovski (L_1) form distance.

$$L_c(H_1, H_2) = \frac{1}{L} \sum_{i=1}^{L} \frac{||H_1(i) - H_2(i)||^c}{M} \tag{4.8}$$

where H_1, H_2 are two histograms of the same size of L. M is the largest possible magnitude of the shifted coefficients. The coefficients in the histograms are shifted to make the minimum energy level to be 0. The reason to choose the distance metric is not introduced in the paper.

Lu et al. [56] extract a vector quantization index histogram from the DCT coefficients. The 64 DCT coefficients in each 8×8 block are divided into four groups as shown in Figure 4.6. For each color channel (Y, Cb, Cr), 4 codebooks are trained

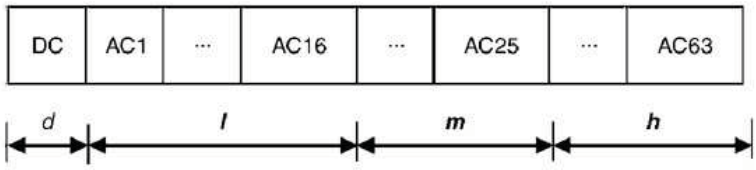

Figure 4.6: Partition of DCT coefficients in the vector quantization index histogram

from a randomly selected training image set. After obtaining the above 12 codebooks, images in the data set are processed in the same method as the training images. The DCT sequences are divided into 12 groups, and encoded by the corresponding codebooks. The indexes of the DCT coefficients from each codebook are jointly put together as the quantization index histograms.

Lu et al. [57] approximate the color and texture feature in the pixel domain directly from the DCT coefficients. The color features of the image are calculated directly from the DCT coefficients by partial decoding the JPEG image. Each 8×8 DCT block is divided into 4 sub-blocks as shown in Figure 4.7. The average color values of each block is denoted as $M_{11}, M_{12}, M_{21}, M_{22}$. The value of $M_{11}, M_{12}, M_{21}, M_{22}$

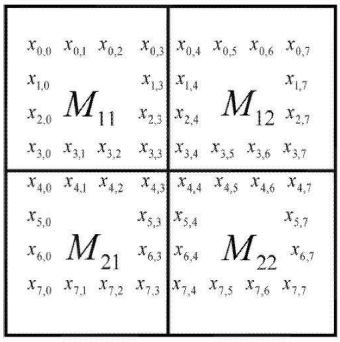

Figure 4.7: 4 sub-blocks for each DCT block

are approximated through the four upper left coefficients in the 8×8 DCT block. The texture information is a vector extracted from selective DCT coefficients from 6 groups. Group 1 is the DC coefficient. Group 2 and 3 are the frequency information. Group 4, 5 and 6 are the vertical, horizontal and diagonal direction information. The mean and the standard deviation of all coefficients in each group are extracted as the texture features.

The Euclidean distance is used in [56, 57]. The Euclidean distance is also known as L_2 measure, and is defined as in Equation (1.1) in Section 1.3.3. Note that no reason is given for choosing the distance measure for the proposed features.

Feng et al. [23] extract a set of moments directly from DCT coefficients without full decompression or the inverse DCT step. They show that the mean μ_1 and variance σ_1^2 for each $N \times N$ DCT block can be calculated directly by Equation (4.9) and (4.10).

$$\mu_1 = \frac{1}{N} C(0,0) \tag{4.9}$$

$$\sigma_1^2 = \frac{1}{N^2} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C^2(u,v), (u,v) \neq (0,0) \tag{4.10}$$

where $C(u,v)$ is the DCT coefficient at position $(u,v), 0 \leq u,v \leq 7$. N is 8 for JPEG images. The equations show that the mean is derived from DC coefficients, and the variance is derived from AC coefficients. The joint space of mean and variance is quantized into a space of size 4×7, where the mean is divided into 4 sections, and the variance into 7 sections. Then the image feature is extracted as an index histogram with each bin indicating the frequency of the joint mean-variance in the images. The

distance measure is based on a weight distance transform (WDT) technique. Each item in the histogram is projected into a distance image. The distance between the elements in two histograms is calculated by the common area between the two distance images. As the operations of the multiplication in the L_2 measure are avoided, this distance measure is more efficient. However, the choice of the distance measure is to improve the efficiency. Whether the measure is suitable for the defined features is not discussed in the work.

To make effective use of features extracted from DCT coefficients, some other works focus on designing more intuitive distance metrics for DCT features. For instance, H. Yu [107] introduces a Q-metric for the DCT coefficients to measure the image similarity. A sub-image including all DCT coefficients at each position in a 8×8 block is generated and the wavelet transformation is applied. While calculating the distance between a query and a target image, the Q-metric counts the total number of the coefficients which are higher than a predefined threshold in the corresponding sub-images from both the query and target images. The overall distance is a weighted summation of the counts from all sub-images. The Q-metric gives a measurement on how many significant coefficients in two images are in common. However, how to decide a coefficient is significant depends on the heuristic thresholds.

Observe that in the works reviewed above, the design of the features, and more importantly the associated distance measures for image matching, are often two completely separate decisions. Thus, the design involves a fair amount of heuristic decisions. In our work, we define the features as the DCT coefficients first. By formulating the retrieval process as a hypothesis testing problem, we derive the theoretical sound distance metric for retrieval using the DCT features. The DCT2KL algorithm reduces the dependence on heuristics by treating image retrieval as a hypothesis testing problem. The choice of the features and the distance measure are decided as one decision.

4.3 Experiments

We evaluate our proposed DCT2KL method on two image data sets: the Corel1K data set [11, 102] and the UCID data set [81]. The Corel1K data set consists of 1000 images in 10 classes. We run 1000 queries with each image serving as a query

image. Two images in this set are considered relevant when they are in the same class, and irrelevant otherwise. The UCID data set consists of 1338 color images. It has a ground truth file with 262 queries that can be used to evaluate retrieval results.

4.3.1 Comparison to Other Features

We compare our proposed retrieval scheme with the one using other features such as the RGB histograms [93], the RGB GMMs [38], and the low frequency DCT histograms [49]. The RGB histograms are three 256-bin histograms extracted from R, G, and B channel respectively. Similarities between two histograms are calculated by the KL divergence. The similarities from three channels are summed together as the total similarity between the RGB histograms of two images. RGB GMMs feature is a multi-dimensional color feature, which provides a compact representation of RGB three-dimensional data. The RGB GMM feature of an image is a GMM trained from the RGB three-dimensional data. The EMass algorithm [38] is applied for training GMMs. Similarities between two GMMs are measured by the unscented transformed (UT) based approximation distance measure [28] in Equation (3.32). The low frequency histograms are extracted as in [49]. The distance between low frequency histograms are measured by Equation (4.8).

As suggested in Section 4.2.3, the DCT2KL algorithm in Table 4.9 uses $K = 13$, which includes the 4 sequences from DC and the first 9 AC sequences. The same K is used in all these experiments.

Table 4.9 shows the MAP results of the retrievals by using different features and methods. The MAP is define in Equation (1.14) in Section 1.4.2.

Table 4.9: MAP comparison of the RGB hist, the RGB GMMs, the low frequency DCT histograms, and the DCT2KL algorithm

Method	Corel1K	UCID
RGB Hist	0.398	0.505
RGB GMM	0.481	0.535
Low freq. DCT	0.538	0.617
DCT2KL	0.657	0.690

Table 4.9 shows that our DCT2KL algorithm significantly outperforms the other two color features on both data sets. The DCT coefficients obviously present much

better discriminating capability than the RGB features. Conversion from RGB to YCbCr provides a good separation among the luma and chroma channels. The DCT brings further decorrelation among the coefficients. As the DCT2KL retrieval scheme is derived based on the decorrelation property, this performance gain is consistent with our assumptions.

The DCT2KL method also achieves higher precision than the low frequency DCT histograms. Unlike the low frequency histograms which extract features from groups of DCT coefficients, DCT2KL extracts a histogram from each coefficient separately. According to the DCT coefficients properties, each DCT coefficient can be processed independently without losing information after the decorrelation. Moreover, our distance measure in DCT2KL is derived directly based on the features. Thus, the DCT2KL gains better performance than the low frequency histograms.

4.3.2 Comparison to Other Systems

In order to make comparisons to other retrieval systems which have been evaluated on these two data sets, we also evaluate our scheme by using the same measures adopted for evaluating these corresponding systems.

The left table in Table 4.10 shows the average precisions of top 100 retrieved results using SIMPLIcity [102], CLUE [11], and our DCT2KL scheme for the Corel1K data set. The SIMPLIcity is a region-based image retrieval system using wavelet-based features. The CLUE system combines the same region-based features with a clustering algorithm to boost the retrieval precision. The region-based systems involve image segmentation and region-based matching, which both are computational intensive. Our DCT2KL algorithm is not only much simpler and more efficient, but also retrieves with higher precision than these two systems.

CVPIC is an image feature which is based on block color co-occurrence matrix and pattern histogram, and it achieves state-of-the-art performance on the UCID data set [81]. The right table in Table 4.10 compares our retrieval approach to the CVPIC [80] feature on the UCID data set. The performance is measured by AMP as in [80]. The retrieval performance of DCT2KL is slightly better than the CVPIC feature in this experiment.

It is difficult to make direct empirical comparison with other related works in

Table 4.10: Comparison to other systems

Method	Top100
SIMPLIcity	0.477
CLUE	0.538
DCT2KL	0.604

Method	AMP
CVPIC	94.23
DCT2KL	95.76

DCT domain largely due to the inaccessibility of most image data sets used in these works. In an exception, Lu et al. [57] report the retrieval results by a joint color-texture DCT feature for the Corel1K data set in a Precision-Recall (P-R) figure, which is the average of 5 query images randomly picked from each class. We conduct the experiment in the same way as in [57]. Figure 4.8 clearly shows the superior performance by DCT2KL compared to their DCT-based feature.

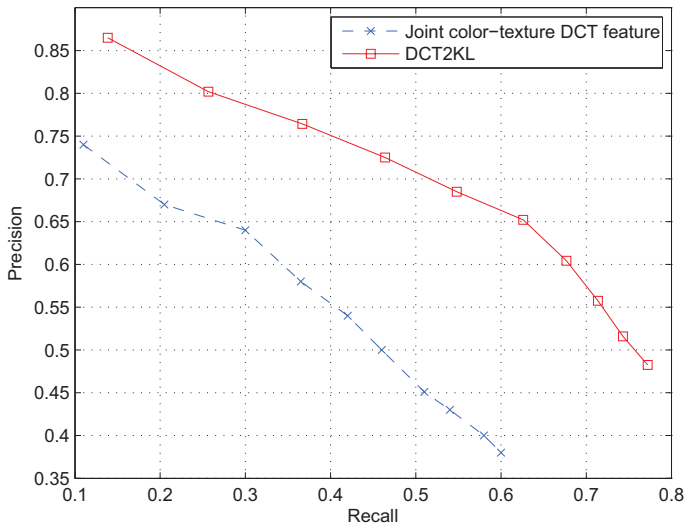

Figure 4.8: Retrieval precision-recall Figure for the Corel1K data set.

4.3.3 The Number K

In view of the energy compaction property of DCT, we investigate the sensitivity of our DCT2KL algorithm to K, that is, the number of DCT coefficients used. Figure 4.9 shows the retrieval MAPs on the two data sets using different K. Note that $K \geq 4$ implying that the four sequences from the DC coefficients are always included, and the AC coefficients are incrementally included according to the zigzag order defined in

the JPEG standard. Since the curves in Figure 4.9 peak around $K = 13$, we see that good performance can be achieved by using DC together with 6 to 10 AC coefficients. This result confirms that the information relevant to image retrieval is concentrated in DC and low frequency AC coefficients.

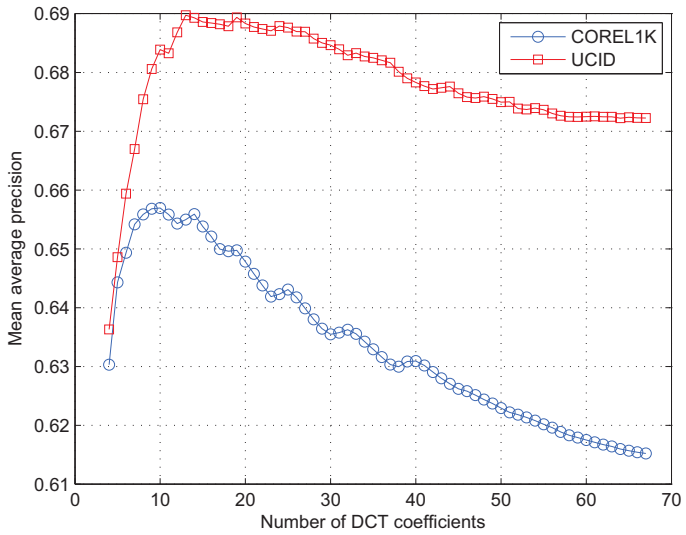

Figure 4.9: MAP by using different numbers of DCT coefficient sequences.

Because of the DCT coefficients have energy compaction property, most energy is in the low index DCT coefficients. Especially for the DCT coefficients restored from JPEG images, the high indexed DCT coefficients are mostly zeros. Images cannot be separated by the information in these coefficients. Thus the performance drops after including more high-indexed DCT coefficients.

4.4 Summary

In this chapter, we have considered a hypothesis testing approach (DCT2KL algorithm) for image matching in the DCT domain. The DCT2KL method uses the DCT coefficients in the YCbCr space restored by partially decoding JPEG images. Assume that these DCT coefficient sequences are emitted from memoryless sources that are independent of each other. For each target image, we form a hypothesis that the DCT coefficient sequences of the query image are emitted from the same sources as

the target image. The DCT2KL algorithm is then derived to identify the most probable hypothesis, which in turn gives the most relevant target image, by minimizing the KL divergence between the query image and the target images. Experimental results on both the Corel1K and UCID data sets show that our DCT2KL algorithm consistently outperforms related systems, often by a wide margin.

In our work, the DCT2KL algorithm is derived directly from the hypothesis testing formulation. The KL divergence is an optimal distance measure for the DCT coefficient features. Although our retrieval experiments are conducted on a small image data set, the algorithm is theoretically ensured to have the same good performance on larger data sets.

In order to further improve the retrieval performance, the DCT2KL scheme can be extended to combine with other kinds of features. The design of such combination scheme and the corresponding features to be included is investigated in the next chapter.

Chapter 5

Feature Combination

In this chapter, we show that the performance of image matching can be improved by combining different features. We provide a scheme to combine different features and methods. The performance of our proposed methods is compared before and after the combination on the two image data sets, Corel1K and UCID data sets. In addition, we also compare our results with the benchmark results by Deselaers et al. [16, 19]. They provide quantitative retrieval results of different features on these two data sets. We then show several image retrieval cases for the Corel1K data set to demonstrate that how our methods and the combination scheme affect the retrieval results.

In order to show the performance of our methods on large-scale image data sets, we provide initial experimental results of our methods on the IAPR TC-12 image data set, which contains 20,000 generic images. The retrieval results of the topics in the ImageCLEF 2007 photographic retrieval task [33] by using our methods are compared with related results on these topics on the same data set. To further improve the retrievals, we combine our methods with three MPEG-7 image descriptors. Several retrieval cases on the IAPR TC-12 data set are provided as well.

5.1 Combining Scheme

The LTP histograms, the RGB GMMs, and the DCT2KL method can be combined to provide more effective retrieval results. The LTP histograms represent the spatial relationships among the color levels of neighboring pixels. These spatial relationships provide one type of the texture information in images. The RGB GMMs are features with color information. Retrievals by the RGB GMMs only return images similar in RGB colors. The DCT2KL method contains both the color information (DC coefficients) and the texture information (AC coefficients) in the YCbCr color space. Thus these three methods contain the different information in different color spaces. The combinations of the LTP histograms, the RGB GMMs, and the DCT2KL are

able to retrieve images that are similar to the query image in terms of both the color and the texture.

The combination is made in two steps. First, the distance between the query image and all target images are calculated by each method separately. Then the distance by each method is normalized to a value between 0 and 1 by the maximum distance, and summed together as the combined distance. The joint distance measure is thus defined as:

$$D_c(Q, T_i) = \sum_{j=1}^{k} \left(\frac{D_j(f_j, g_{ij})}{\max_j D_j(f_j, g_{ij})} \right), i = 1, \ldots, n \tag{5.1}$$

where Q is query image, and T_i is the i-th target image. Each query is performed by using k types of methods. The corresponding features for the j-th method are $f_j, j = 1, \ldots, k$ from image Q, and $g_{ij}, j = 1, \ldots, k$ from image T_i. D_c is the combined distance measure. D_j is the distance measure for the j-th feature. After calculating the distance $D_c(Q, T_i), i = 1, \ldots, n$ for each target images T_i, they are sorted by the distance to provide the final retrieval list.

5.2 Experimental Results for the Corel1K and the UCID Data Sets

5.2.1 Results using Different Features from the Literature

The Corel1K and the UCID data sets are two widely used image data sets for CBIR. Deselaers et al. [16, 19] provide extensive experimental results of image retrieval using different features on these two data sets. Their work is initially started with the organization of the ImageCLEF from 2005 to 2008. The performance of retrievals by using a large collection of different features are provided and discussed extensively. By comparing to their results, we are able to confidently judge the effectiveness of different methods.

The features that have been experimented on these two data sets are: color histograms, LF SIFT global search, LF patches histogram, invariant feature histogram, MPEG-7: scalable color, LF patches signature, Gabor histogram, 32 × 32 image, MPEG-7: color layout, Xx32 image, Tamura texture histogram, LF SIFT signature, gray value histogram, LF patches global, MPEG-7: edge histogram, invariant feature histogram (relational), Gabor vector, and global texture feature. To see the feature

details refer to reference [19].

The performance of each feature is compared by two metrics, Error Rate (ER) and MAP. The ER metric is defined as $1-P(1)$, where $P(1)$ stands for the precision when the first relevant image, is retrieved. The ER shows that how quick the first relevant image can be found. The MAP is defined in Equation (1.14), which is the mean of the precision at each relevant image is retrieved. The MAP metric measures the average performance of multiple retrieved images. To evaluate the retrievals only requiring a single relevant image, the ER metric is more suitable. To evaluate multiple retrieval queries, the MAP metric is more suitable. The retrieval ER and MAP results by using these features in [19] are shown in Table 5.1.

Table 5.1: ER and MAP [%] for each of the features for the Corel1K and the UCID data sets [19]

features	Corel1K		UCID	
	(ER)	(MAP)	(ER)	(MAP)
color histogram	16.9	50.5	51.5	43.3
LF SIFT global search	37.2	38.3	31.7	62.5
LF patches histogram	17.9	48.3	58.0	37.5
LF SIFT histogram	25.6	48.2	50.4	44.7
inv. feature histogram(monomial)	19.2	47.6	53.8	41.6
MPEG-7: scalable color	25.1	46.7	60.7	37.9
LF patches signature	24.3	40.4	68.7	27.6
Gabor histogram	30.5	41.3	74.1	22.3
32x32 image	47.2	37.6	82.8	14
MPEG-7: color layout	35.4	41.8	75.2	21.7
Xx32 image	55.9	24.3	83.2	13.9
Tamura texture histogram	28.4	38.2	63.4	33.2
LF SIFT signature	35.1	36.7	58.4	34.1
gray value histogram	45.3	31.7	86.6	11.8
LF patches global	42.9	30.5	63.4	30.3
MPEG-7: edge histogram	32.8	40.8	69.9	25.2
inv. feature histogram(relational)	38.2	34.9	83.2	14.4
Gabor vector	65.5	23.7	95.8	4.7
global texture feature	51.4	26.3	95.4	6.7

From these results, we observe that the best performance for the Corel1K data set is achieved by the color histogram with ER (16.9%) and MAP (50.5%). The best performance for the UCID is using the LF SIFT global search with ER(31.7%)

and MAP(62.5%). Deselaers et al. [19] also show the retrieval MAP results for the Corel1K data set by combining different features. The selective features and the corresponding retrieval MAPs are listed in Table 5.2. The best performance is improved to ER(11.6%) and MAP(55.7%).

Table 5.2: ER and MAP [%] by combining features for the Corel1K data set [19]

features	ER(%)	MAP(%)
color histograms	16.9	50.5
+ global texture	15.7	49.5
+ Tamura histograms	13.7	51.2
+ thumbnails	13.7	53.9
+ patch histograms	**11.6**	**55.7**

5.2.2 Results using Our Methods

In order to compare to the reported results in Table 5.1 and 5.2, we list the ER and MAP results by using our proposed methods in Table 5.3. Note that the results in this table use the same leave-one-out configuration that the query image is removed from the target sequence as in the Table 5.1 and 5.2.

Table 5.3: ER and MAP [%] for each of our proposed method for the Corel1K and the UCID data sets

features	Corel1K		UCID	
	(ER)	(MAP)	(ER)	(MAP)
LTP histograms	12.0	48.8	65.2	26.7
RGB GMMs	14.3	47.8	41.8	49.6
DCT2KL	**7.8**	**62.7**	**38.2**	**52.6**

Our DCT2KL algorithm significantly outperforms the reported state-of-the-art results in Table 5.1 for the Corel1K data set. Compared to the results in Table 5.2, which combines 5 features, the DCT2KL method still outperforms their results by both the ER and MAP. For the UCID data set, our performance is above all of the features, except the LF SIFT global search. The ground truth for the UCID shows that the retrievals require information at the object level. The SIFT features used in the LF SIFT global search provide such information. Thus their performance is better.

The ER and MAP of retrievals by using the combination of different features and methods are shown in Table 5.4.

Table 5.4: ER and MAP [%] for combining the proposed features and methods for the Corel1K and UCID data sets

features	Corel1K		UCID	
	(ER)	(MAP)	(ER)	(MAP)
LTP+RGB GMMs	7.75	58.2	47.8	51.1
LTP+DCT2KL	5.45	65.3	37.1	54.3
LTP+DCT2KL+GMMs	**5.18**	**67.90**	**35.7**	**56.1**

The performance on these two data sets, especially the Corel1K is improved again after the combination. The LTP histograms describe the relationships between a pixel and its eight neighboring pixels, which provide the texture information. The RGB GMMs provide the color information in the RGB color space. The DCT2KL method provides color and texture information in the DCT domain in the YCbCr color space. These combinations take the advantages of both the texture and color information in different color spaces, thus achieve better performance than the performance of each feature by itself. The improvement for the UCID data set is not as significant as the retrievals for the Corel1K, because the retrievals for the UCID data set require more object-level information.

5.2.3 Retrieval Examples

We provide three groups of examples to show the performance of the LTP histograms, the GMMs trained by the EMass algorithms, the DCT2KL methods, and the combination of them for the Corel1K data set. For each query image in the Corel1K data set, there are 100 relevant images to be retrieved. By showing a long list of retrieved images, we are able to compare and observe how each of method affects the retrieval performance.

The different query images in each group of the examples are: one in the “Flower” category, one in the “Africa” category, and one in the “Mountain” category. The top 32 retrieved images that are similar to the given query image are listed in these examples.

The query image of the first group of examples in Figures 5.1, 5.2, 5.3, and 5.4 is in the *flower* category. Figure 5.1 shows the query image and the retrieved images ranking from 1 to 8. The query image is the one in the first column. The second column is the retrieved images by using the LTP histograms. The third column is the retrieval images by using the RGB GMMs. The fourth column is the retrieved images by using the DCT2KL method. The fifth column is the retrieved images by using the combination of all three methods. The positive retrieved images are marked by red rectangles, and the negative retrieved images are marked by blue dotted rectangles. Figures 5.2, 5.3, and 5.4 list the retrieved images ranking from 8 to 16, 17 to 24, and 25 to 32 for each method and their combination respectively in the same manner as in Figure 5.1.

The retrieval by using the LTP histograms returns 24 positive images. The LTP histograms retrieve images with similar spatial relationships in neighbor pixels without the color information. Thus, we cannot observe the color similarity in the retrieved target images, e.g., the white flower image at rank 6. The similarity is in a way that two main types of spatial relationships, with one part as non-smooth area, and the other as smooth background. The retrieval by using the RGB GMMs returns 16 positive images and 8 negative images. The RGB GMMs find images similar to the query image in colors. The query image is a flower in orange color with a dark background. The retrieved images are thus all contain both the orange and the dark colors. The DCT2KL method finds images similar to the query image both in color and texture, as we can see from the results in the column 3. Flower images with different colors are returned by the DCT2KL method. The results show that the DCT2KL method is able to find images similar to the query image very effectively.

The combination results show that using different methods together is a re-ranking process of the retrieved target images. The images with top ranks by all methods are returned first in the combined retrievals. These examples provide an initial impression on how each of our presented methods performs on image matching, and how the combination scheme we apply affect the results.

Another two groups of examples with different query images are shown in Figures 5.5 to 5.12. The retrieved images are shown in the same manner as in the previous examples. These two groups of results show similar retrieval patterns as

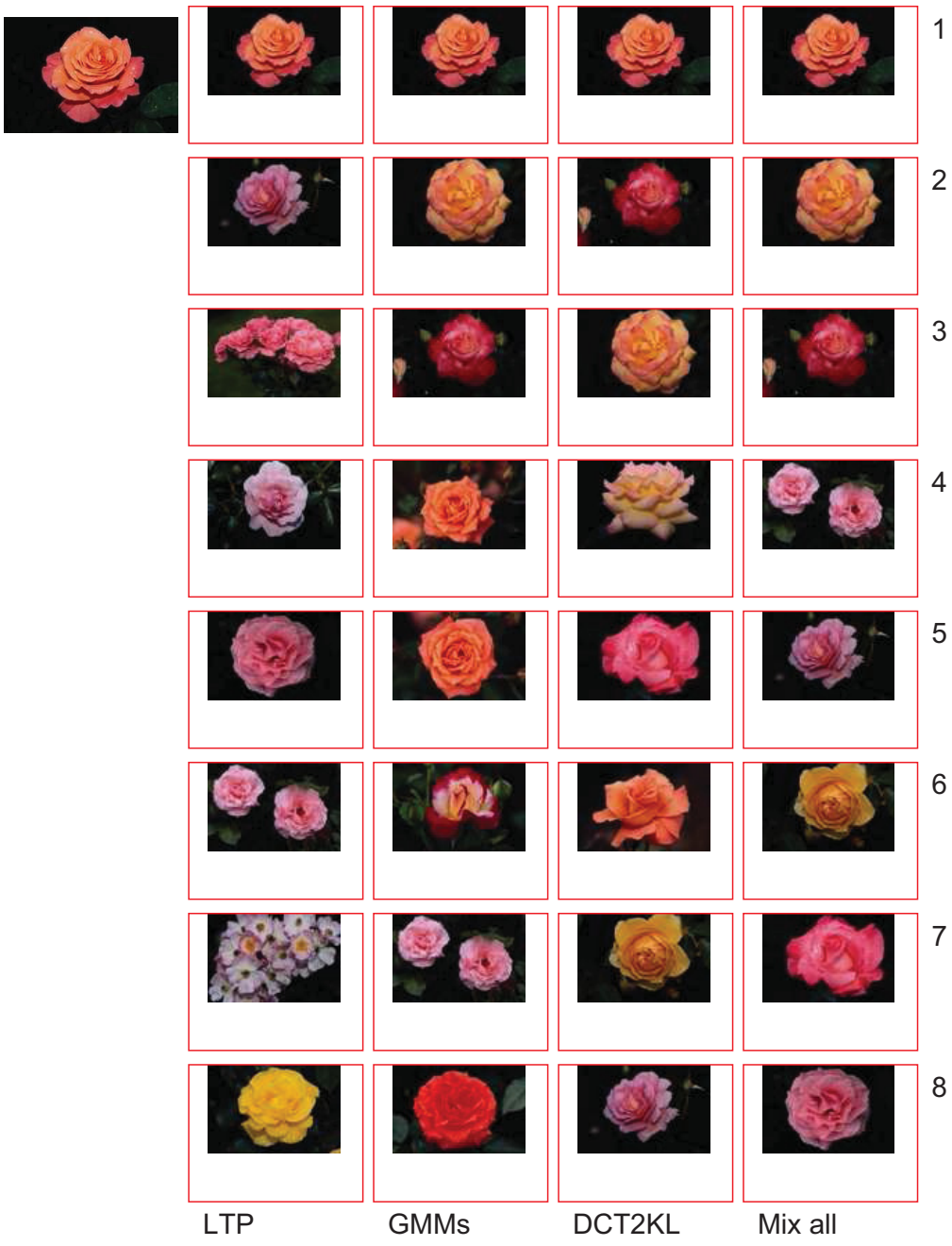

Figure 5.1: Retrieved images ranking from 1 to 8 by using the LTP histograms (column1), the GMMs of RGB (column2), the DCT2KL method (column3), and their combination (column4) with a query image in the *Flower* category. (Red rectangle: positive; blue dotted rectangle: negative)

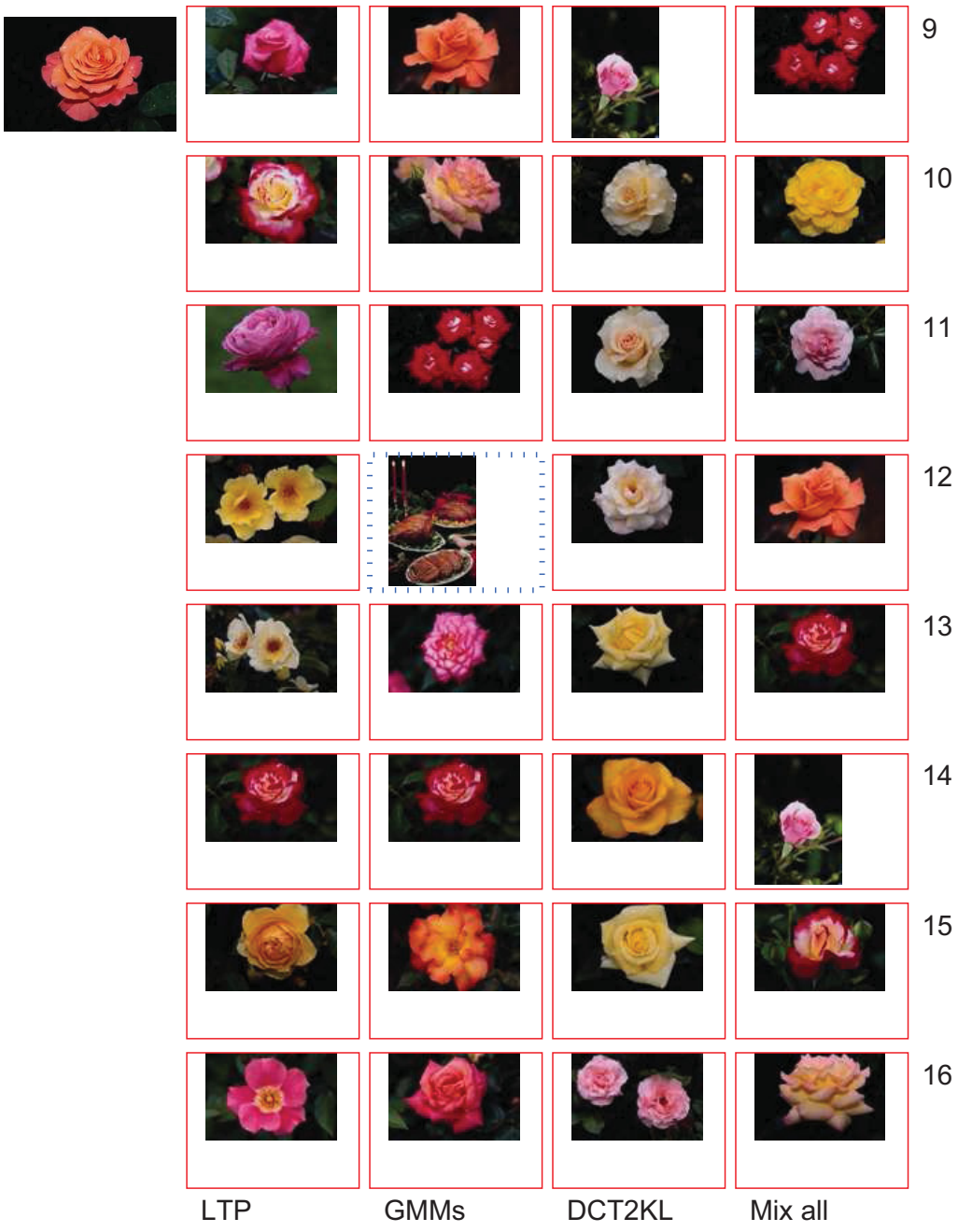

Figure 5.2: Retrieved images ranking from 9 to 16 by using the LTP histograms (column1), the GMMs of RGB (column2), the DCT2KL method (column3), and their combination (column4) with a query image in the *Flower* category. (Red rectangle: positive; blue dotted rectangle: negative)

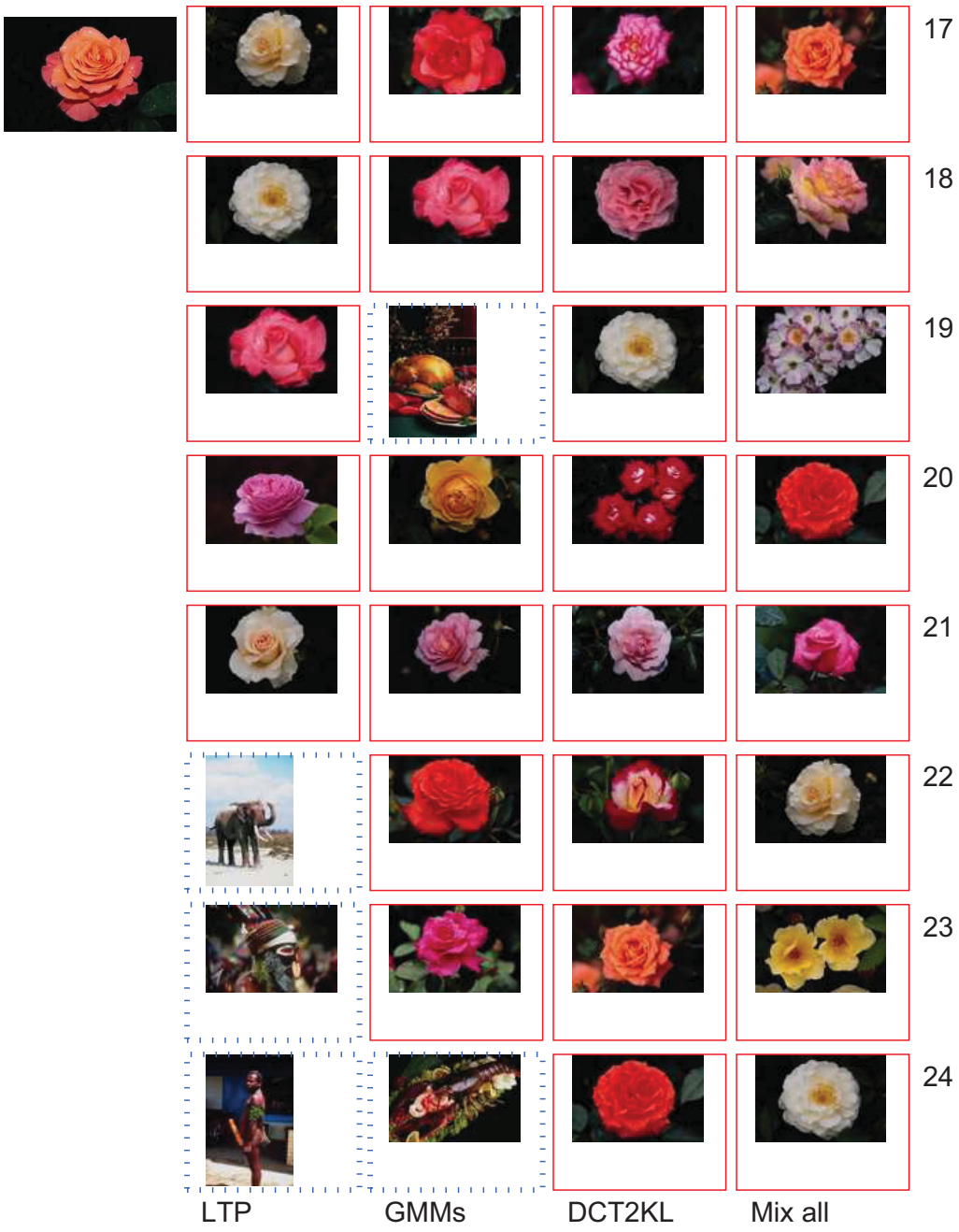

Figure 5.3: Retrieved images ranking from 17 to 24 by using the LTP histograms (column1), the GMMs of RGB (column2), the DCT2KL method (column3), and their combination (column4) with a query image in the *Flower* category. (Red rectangle: positive; blue dotted rectangle: negative)

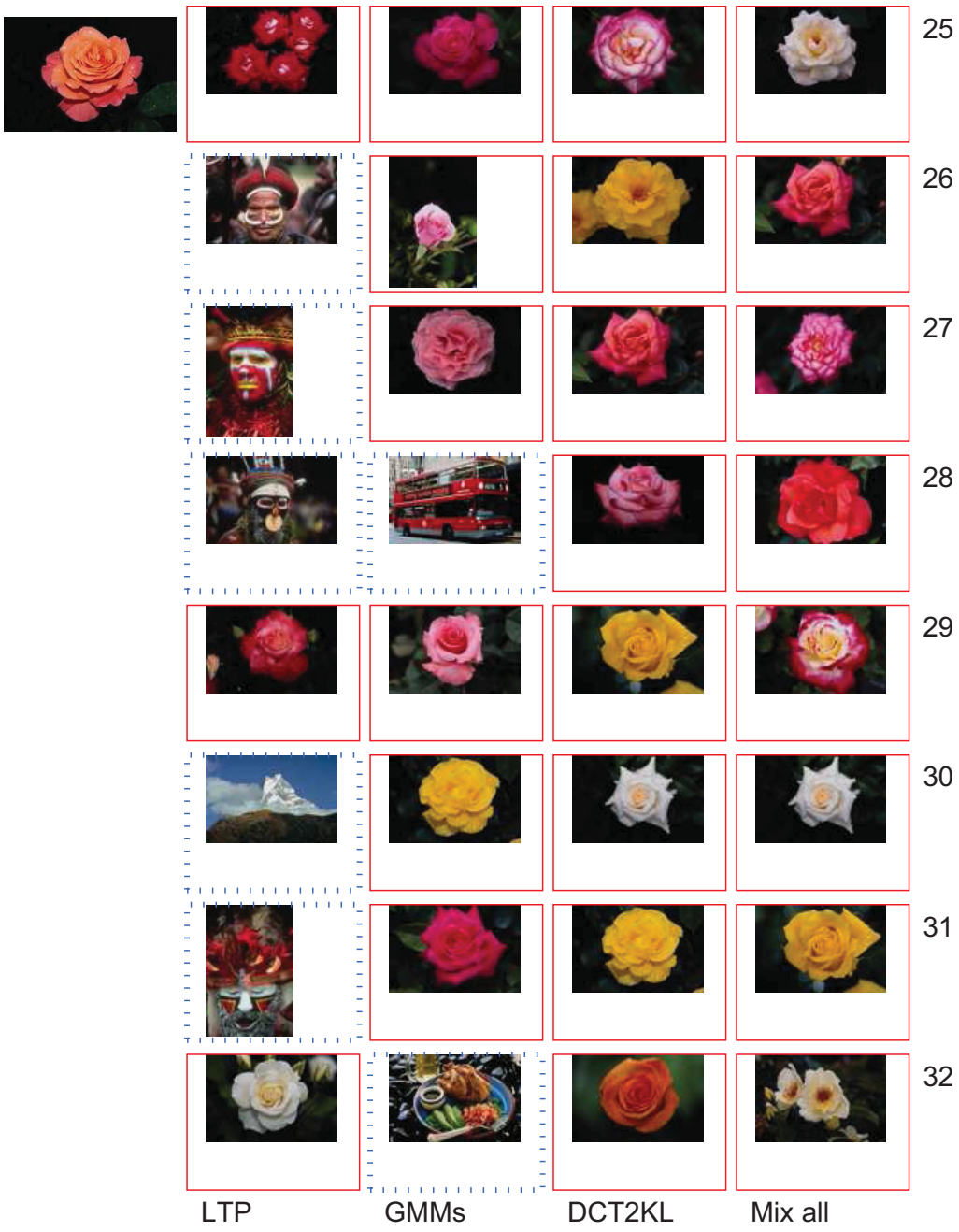

Figure 5.4: Retrieved images ranking from 25 to 32 by using the LTP histograms (column1), the GMMs of RGB (column2), the DCT2KL method (column3), and their combination (column4) with a query image in the *Flower* category. (Red rectangle: positive; blue dotted rectangle: negative)

in the previous group of examples in Figures 5.1 to 5.4. Retrievals using the LTP histograms find images with similar texture to the query image. Retrievals using the RGB GMMs find images with similar colors. The DCT2KL method is able to find images that are similar to the query images by both the color and texture. By combining these methods, the retrieved target images are more precise. These three groups of experimental results initially demonstrate how our proposed methods and their combination positively affect retrievals.

5.3 Experimental Results on the IAPR TC-12 Data Set

In this section, we discuss how our methods perform on large scale image data sets. The image collection of the IAPR TC-12 benchmark consists of 20,000 images collected from different sources. These images are all natural still images, including different sports and actions, photographs of people, animals, cities, landscapes and many other aspects of contemporary life [33, 34]. Some example images are shown in Figure 1.6 in Section 1.4.1.

5.3.1 ImageCLEF Photographic Retrieval Task and Results

The ImageCLEF 2007 Photographic Retrieval Task [33] provides 60 queries which are shown in Table 5.5. Three sample images are listed for each topic. For image matching by visual content, these sample images are used as query images. It is easy to notice that these query topics are very difficult as the topics are described in very specific details. For example, for the topic 2 "church with more than two towers", images contain churches with one tower is a negative match for the topic.

The retrievals are evaluated using the MAP, precision at rank 20 (P20), the geometric MAP (GMAP) to test system robustness, and the binary preference (bpref) measure which is an indicator for the completeness of relevance judgments [33]. The GMAP is defined by TREC 2004, which provides a geometric mean of per-topic average precision in Equation (5.2).

$$\text{GMAP} = \sqrt[n]{\prod_{n} AP_n} \tag{5.2}$$

where AP is average precision. n is typically as 50 for TREC tasks.

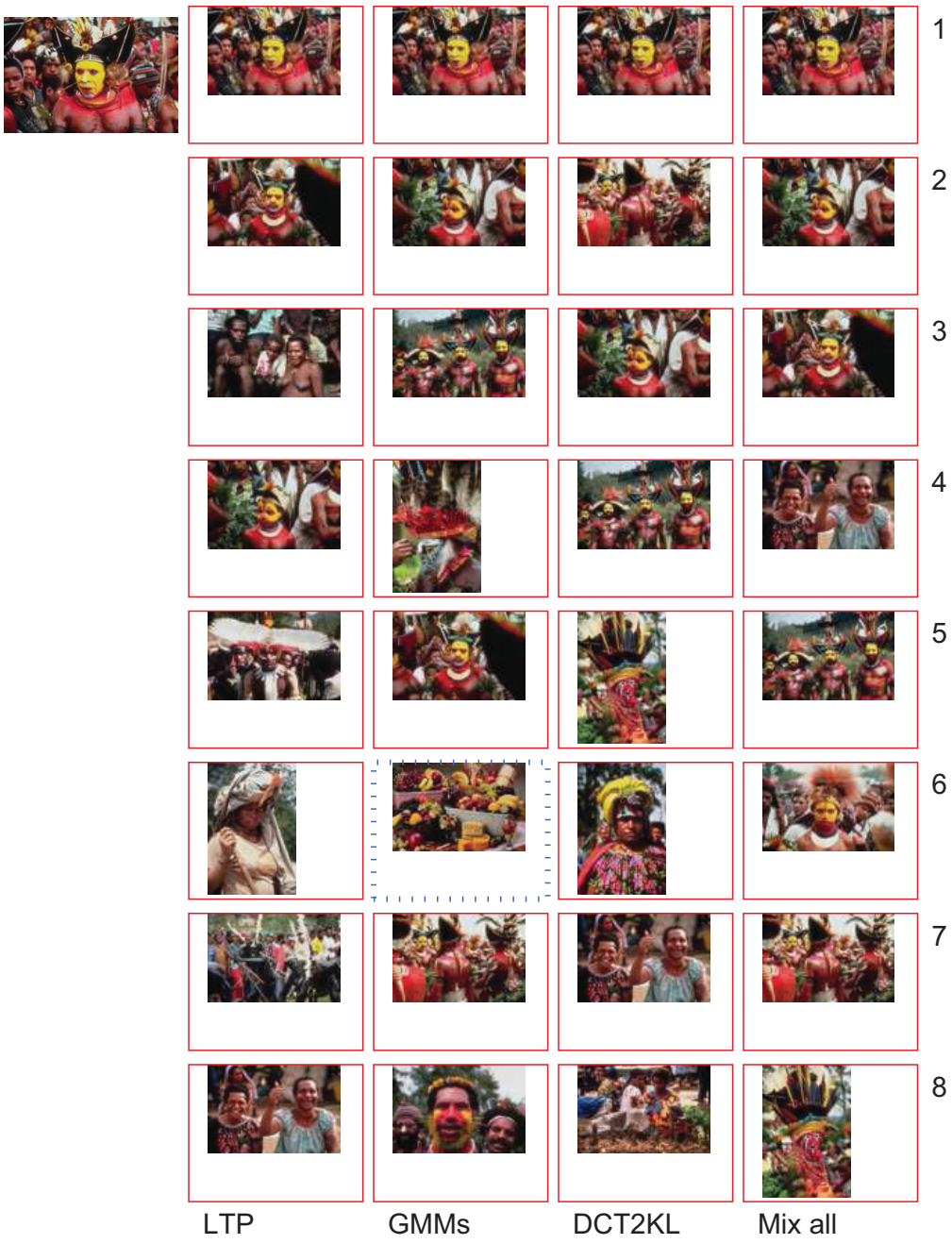

Figure 5.5: Retrieved images ranking from 1 to 8 by using the LTP histograms (column1), the GMMs of RGB (column2), the DCT2KL method (column3), and their combination (column4) with a query image in the *Africa* category. (Red rectangle: positive; blue dotted rectangle: negative)

Figure 5.6: Retrieved images ranking from 9 to 16 by using the LTP histograms (column1), the GMMs of RGB (column2), the DCT2KL method (column3), and their combination (column4) with a query image in the *Africa* category. (Red rectangle: positive; blue dotted rectangle: negative)

Figure 5.7: Retrieved images ranking from 17 to 24 by using the LTP histograms (column1), the GMMs of RGB (column2), the DCT2KL method (column3), and their combination (column4) with a query image in the *Africa* category. (Red rectangle: positive; blue dotted rectangle: negative)

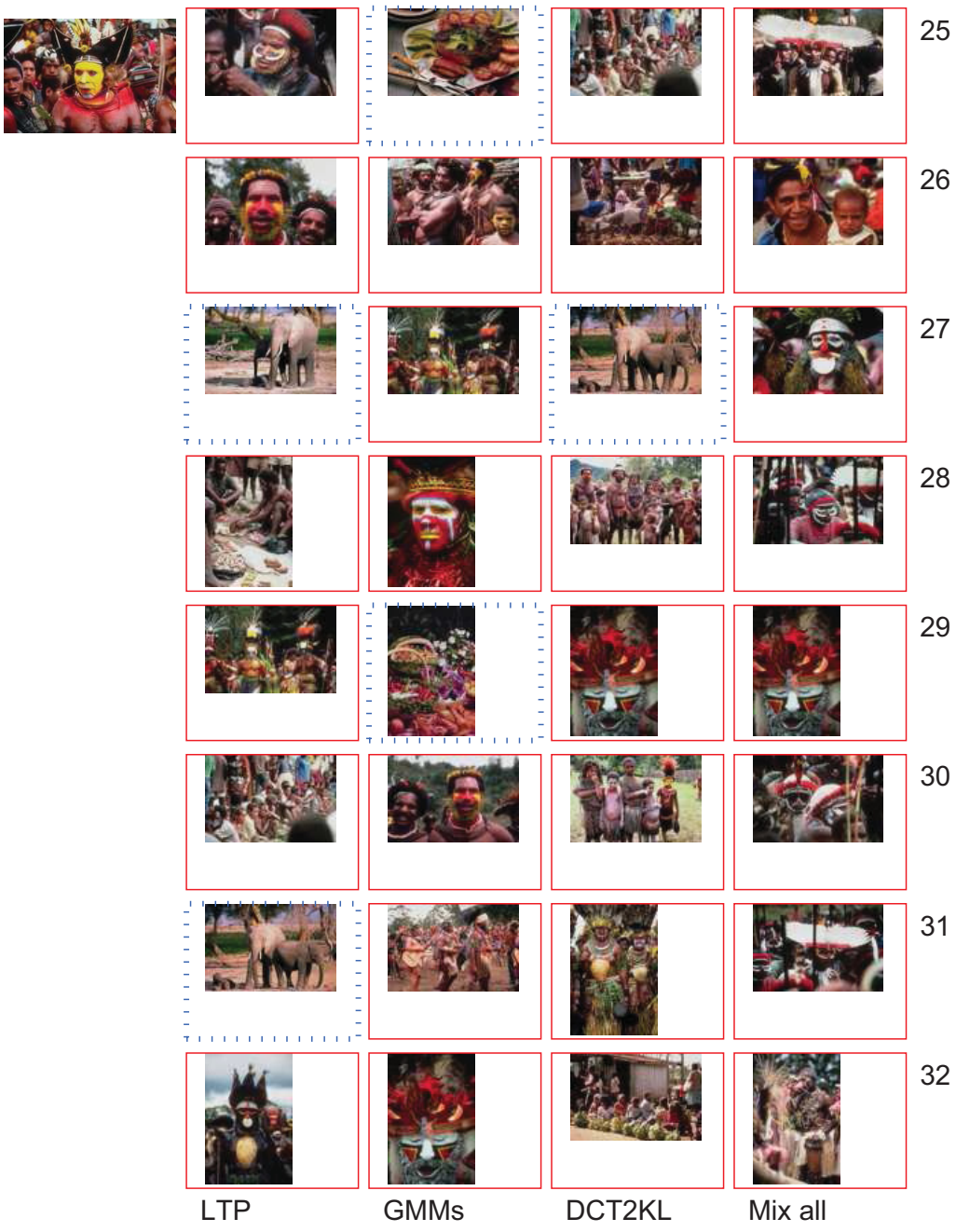

Figure 5.8: Retrieved images ranking from 25 to 32 by using the LTP histograms (column1), the GMMs of RGB (column2), the DCT2KL method (column3), and their combination (column4) with a query image in the *Africa* category. (Red rectangle: positive; blue dotted rectangle: negative)

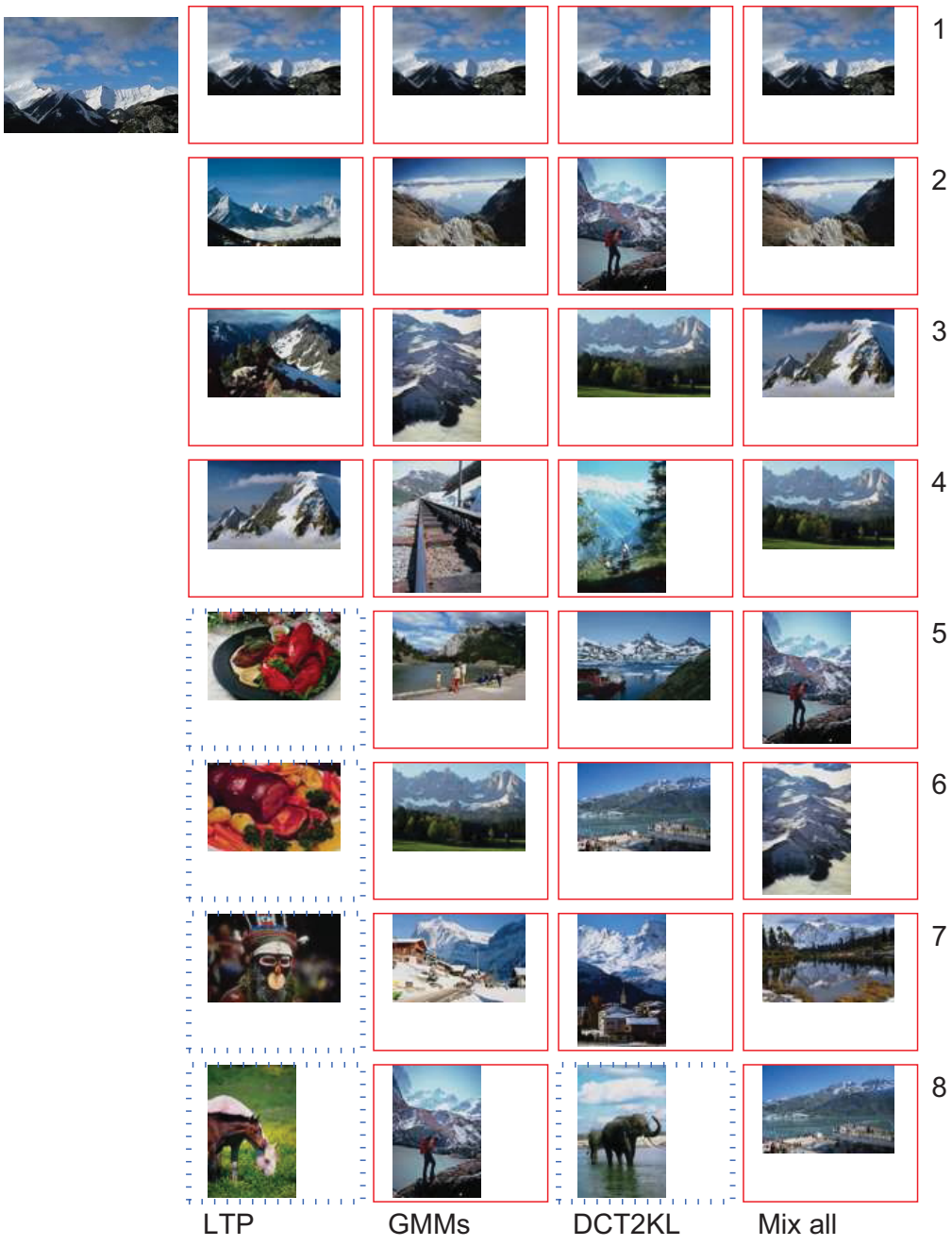

Figure 5.9: Retrieved images ranking from 1 to 8 by using the LTP histograms (column1), the GMMs of RGB (column2), the DCT2KL method (column3), and their combination (column4) with a query image in the *Mountain* category. (Red rectangle: positive; blue dotted rectangle: negative)

Figure 5.10: Retrieved images ranking from 9 to 16 by using the LTP histograms (column1), the GMMs of RGB (column2), the DCT2KL method (column3), and their combination (column4) with a query image in the *Mountain* category. (Red rectangle: positive; blue dotted rectangle: negative)

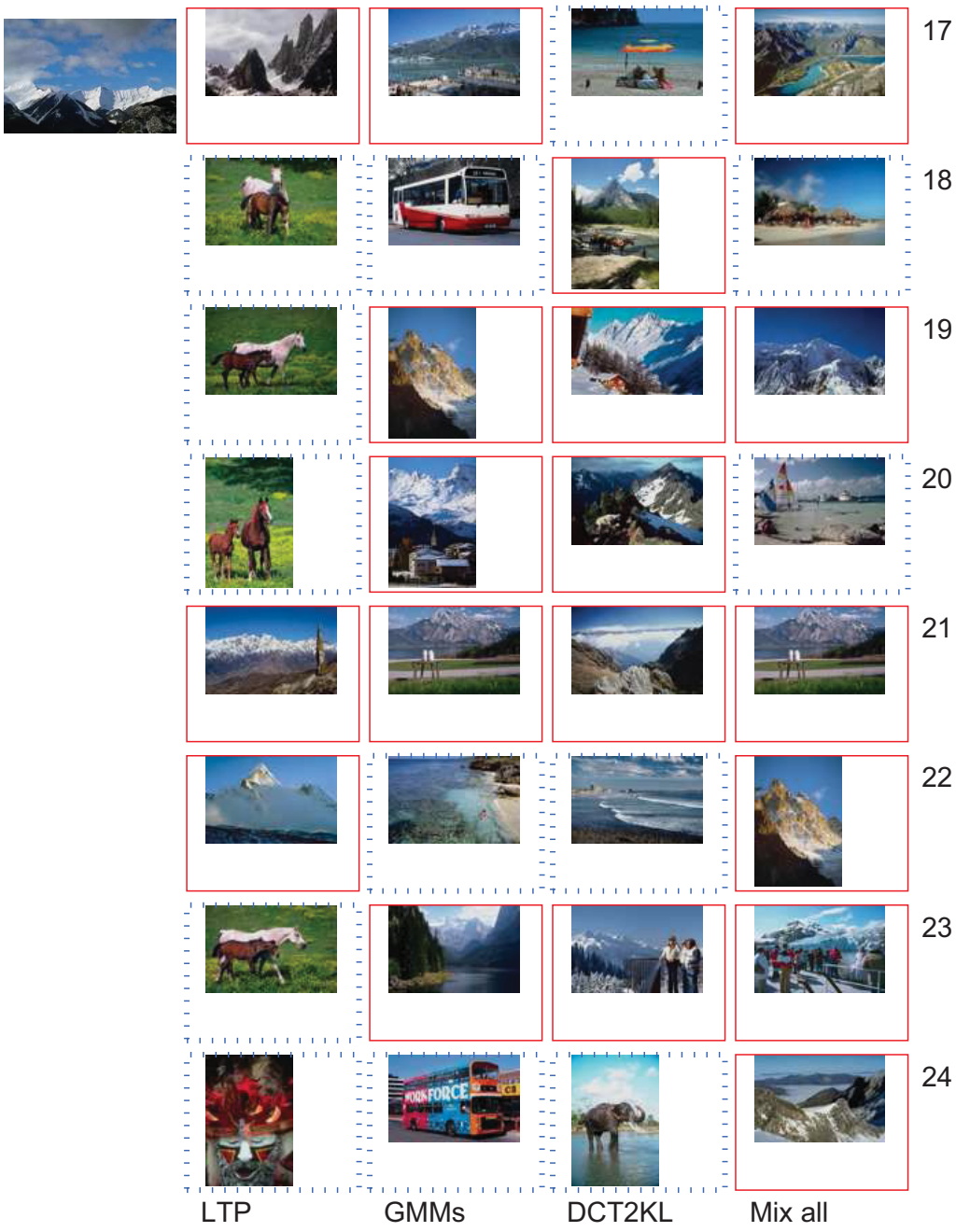

Figure 5.11: Retrieved images ranking from 17 to 24 by using the LTP histograms (column1), the GMMs of RGB (column2), the DCT2KL method (column3), and their combination (column4) with a query image in the *Mountain* category. (Red rectangle: positive; blue dotted rectangle: negative)

Figure 5.12: Retrieved images ranking from 25 to 32 by using the LTP histograms (column1), the GMMs of RGB (column2), the DCT2KL method (column3), and their combination (column4) with a query image in the *Mountain* category. (Red rectangle: positive; blue dotted rectangle: negative)

Table 5.5: ImageCLEF photo 2007 query topics

ID	Topic Title	ID	Topic Title
1	accommodation with swimming pool	31	volcanoes around Quito
2	church with more than two towers	32	photos of female guide
3	religious statue in the foreground	33	people on surfboards
4	group standing in front of mountain landscape in Patagonia	34	group pictures on a beach
5	animal swimming	35	bird flying
6	straight road in the USA	36	photos with Machu Picchu in the background
7	group standing in salt pan	37	sights along the Inca-Trail
8	host families posing for a photo	38	Machu Picchu and Huayna Picchu in bad weather
9	tourist accommodation near Lake Titicaca	39	people in bad weather
10	destinations in Venezuela	40	tourist destinations in bad weather
11	black and white photos of Russia	41	winter landscape in South America
12	people observing football match	42	pictures take on Ayers Rock
13	exterior view of school building	43	sunset over water
14	scenes of footballers in action	44	mountains on mainland Australia
15	night shorts of cathedrals	45	South American meat dishes
16	people in San Francisco	46	Asian women and/or girls
17	lighthouses at the sea	47	photos of heavy traffic in Asia
18	sport stadium outside Australia	48	vehicle in South Korea
19	exterior view of sport stadia	49	images of typical Australian animals
20	close-up photograph of an animal	50	indoor photos of churches or cathedrals
21	accommodation provided by host families	51	photos of goddaughters from Brazil
22	tennis player during rally	52	sports people with prizes
23	sport photos from California	53	views of walls with asymmetric stones
24	snowcapped buildings in Europe	54	famous television (and telecommunication) towers
25	people with a flag		
26	godson with baseball cap	55	drawing in Peruvian deserts
27	motorcyclists racing at the Australian Motorcycle Grand Prix	56	photos of oxidized vehicles
		57	photos of radio telescopes
28	cathedrals in Ecuador	58	seals near water
29	views of Sydney's world-famous landmarks	59	creative group pictures in Uyuni
30	room with more than two beds	60	salt heaps in salt pan

The bpref provides a measure of whether positive images are retrieved ahead of irrelevant images. The measure is given as in Equation (5.3).

$$\text{bpref} = \frac{1}{R}\sum_{r}\left(1 - \frac{|n \text{ ranked higher than } r|}{\min(R, N)}\right) \tag{5.3}$$

where R is the number of positive images, N is the number of negative images, r is a positive retrieved image, n is one negative image in the first R retrieved negative images. $|n$ ranked higher than $r|$ gives the number of negative images in the first R retrieved negative images that are ranked before the positive image r.

The retrieval results by using only visual features reported by different groups are shown in Table 5.6. For groups with multiple submissions, we only keep their best results by MAP.

Table 5.6: Results by different groups for the ImageCLEF 2007 Photographic Retrieval Task

RK	Group	QE RF	MAP	P20	BPREF	GMAP	REL RETR
1	XRCE	NOFB	0.1890	0.3517	0.2009	0.1016	1708
2	DCU	NOFB	0.1340	0.2658	0.1438	0.0515	1339
3	IPAL	NOFB	0.1204	0.2525	0.1316	0.0483	1330
4	INAOE	QE	0.1200	0.1908	0.1196	0.0401	2000
8	RWTH	NOFB	0.1122	0.2383	0.1222	0.0427	1301
11	MIRACLE	NOFB	0.1079	0.2400	0.1206	0.0319	801
34	HongKong	NOFB	0.0511	0.1442	0.0703	0.0172	883
36	RUG	NOFB	0.0337	0.1050	0.0478	0.0078	724
40	CLAC	NOFB	0.0298	0.1000	0.0584	0.0058	368
45	ImpColl	NOFB	0.0280	0.0917	0.0388	0.0061	511
50	Geneva	NOFB	0.0222	0.0717	0.0336	0.0045	719
52	Budapest	NOFB	0.0138	0.0433	0.0240	0.0019	631
	average		0.068	0.157	0.080	0.022	

The XRCE group uses two image features: grey-level SIFT-like features and color features. First, images are divided into non-overlapped 4×4 blocks. SIFT-like feature and RGB sample mean and deviation color features are extracted from each block. Dimensionality of both features is reduced by PCA method. Gaussian Mixture Models are trained from the reduced dimensionality features as visual vocabularies. For each image, Fisher Kernel based normalized gradient vectors [72] are extracted as image features.

The DCU group uses the Flexible Image Retrieval Engine (FIRE) [17] retrieval system, together with six MPEG-7 features, which are Color Layout, Color Structure, Color Moments, Scalable Color, Edge Histogram and Homogenous Texture for retrieval. These features are all defined in MPEG-7 standard. CLAC group also uses MPEG-7 descriptors as scalable Color, color Layout, and edge Histogram.

The IPAL group uses correlograms, HSV histograms, Canny edge operators, Gabor features, and SIFT features as image features. 12 runs of retrievals are performed by different features using different similarity measure methods, which include tf-idf, SVD, Bag-of-visterm, Integrated Statistic Model (ISM), Hidden Maximum Entropy, Word-SVD, and Word-ISM. The final results combine 12 runs using empirically tuned weights.

The RWTH group uses 9 different image features, which are sparse patch histograms, clustered patch histograms, local and global color descriptors from GIFT[1], global texture features, monomial invariant feature histograms, relational invariant feature histograms, Tamura texture histograms, image thumbnails of 32x32 pixels, and RGB color histograms with 512 bins. Their best result in Table 5.6 is tuned from the retrieval results by IAPR 2006 tasks. Without tuning, their best MAP is 0.0834.

The MIRACLE group uses two publicly available CBIR systems GIFT and FIRE. Their best results are achieved by the combination of the retrieval results from two systems. Similarly, Geneva group uses GIFT alone.

For information of the other systems, please refer to the ImageCLEF 2007 summary [33].

5.3.2 Experimental Results using Our Methods

We perform our experiments in the following way. For each topic, the retrieval tasks provide three query images. We first perform three rounds of retrievals by each query image separately. The distance from every target image to the query topic is decided by the minimal distance among the three queries. Results by our methods are shown in Table 5.7.

From the results in Table 5.6 and Table 5.7, it is easy to see that the image matching on this data set by visual content is a very difficult problem. Compared to

[1]http://www.gnu.org/software/gift/, last visited on March 5, 2010.

Table 5.7: Results by using our methods for the ImageCLEF 2007 Photographic Retrieval Task

method	MAP	P20	BPREF	GMAP	REL RETR
LTPN8S32	0.048	0.1283	0.095	0.0173	898
RGB GMM	0.0724	0.1692	0.0963	0.0171	812
DCT2KL	0.0846	0.198	0.1163	0.0285	1113
DCT2KL+LTP+RGB	0.1074	0.2317	0.1293	0.0374	1137

the results in Table 5.6, our methods are simple yet achieve the MAP much higher than the average score. However, we also notice that our methods can be improved by including more features. As the features in our methods contain the global color and local neighboring spatial relation information, we consider including the features with localized color information and edge information in different color spaces which are complementary to our methods.

We provide some initial experimental results by combining our methods with three MPEG-7 descriptors [60]: edge histogram descriptor (EHD), scalable color descriptor (SCD), and color structure descriptor (CSD). These three descriptors include different types of local information, which is complementary to our features and methods. The EHD captures the distributions of edges in an image. The edges are summed into five different types by their orientations. Images are divided into 16 non-overlapped blocks. Edges in each block are counted and joint together as the descriptor. The SCD is a color histogram in the HSV color space encoded by the Haar wavelet transform. The CSD describes the color distribution and local structure of colors of an image in the HMMD (Hue, Max, Min, Diff) color space [79]. A 8×8 block is moved inside the images, and the number of pixels with each color is summarized as the CSD. The step of moving the block is decided by the size of the images. The distance measure for the EHD, SCD, and CSD is the L_1 measure as suggested in MPEG-7 [79]. To extract the MPEG-7 descriptors, we use the MPEG-7 library developed by Bastan et al. [5]. The retrieval results by each of the MPEG-7 descriptor and by combining with our methods are shown in Table 5.8.

The highest MAP in Table 5.5 is 0.1890 by group XRCE. However, their features are extracted based on the PCA process, which requires making use of the features from all the images in the data set. The next best retrieval result by the MAP is

Table 5.8: Results by using three MPEG-7 descriptors (EHD, SCD, and CSD), and by combining with our three methods for the ImageCLEF 2007 Photographic Retrieval Task

method	MAP	P20	BPREF	GMAP	REL RETR
EHD	0.0328	0.0983	0.0821	0.0097	840
SCD	0.0616	0.1567	0.0953	0.0158	884
CSD	0.0788	0.1958	0.1179	0.0275	1011
DCT2KL+LTP+RGB +EHD+ SCD+CSD	0.136	0.2767	0.1569	0.0556	1394

0.1340 by the DCU group, which uses six MPEG-7 features together with the FIRE system. Without the FIRE system, the MAP achieved by the six MPEG descriptors is 0.1000.

Our MAP result is 0.1360 which is the second best in all these results. We simply combine six matching methods, and the weights for different features and methods are equal. The retrieval results on these tasks also indicate a fact that retrievals should be a combination of different information, including color, texture, edge, structure, global, and local features etc.

5.3.3 Retrieval Examples

In this section, we show three groups of retrieval results on the TC-12 data set in Figures 5.13, 5.14, and 5.15. The retrieval topics for these three groups of examples are: church with more than two towers, people on surf boards, and sunset over water. The query images in each topic are shown as the first three images in each Figure. The retrieved target images ranking from 1 to 32 are shown in the same corresponding Figure as the query images. The positive target images given by the ImageCLEF are marked by red rectangles, and the negative retrieved images are marked by blue dotted rectangles.

In Figure 5.13, the query images are churches with more than two towers. The retrieved target images contain buildings which are similar to the query images. However, as the topic specifically requires the buildings as churches and with more than two towers, the target images with other types of buildings or with churches but only one tower are considered as negative matches. Similarly, in Figure 5.14, the topic requires the images with people on surf boards. All retrieved target images are similar

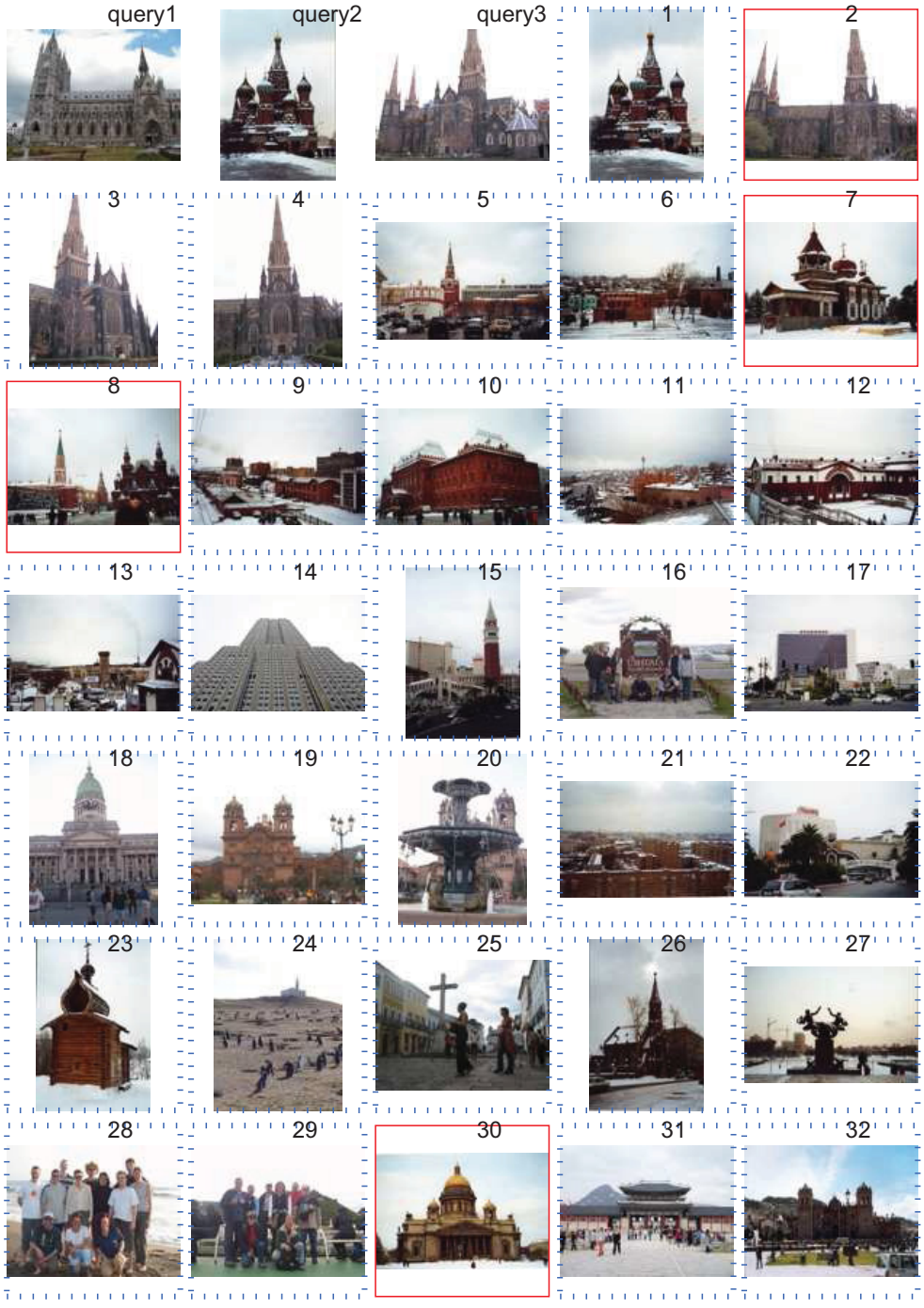

Figure 5.13: Retrieved images for the topic *church with more than two towers* for the IAPR TC-12 data set. (Red rectangle: positive; blue dotted rectangle: negative)

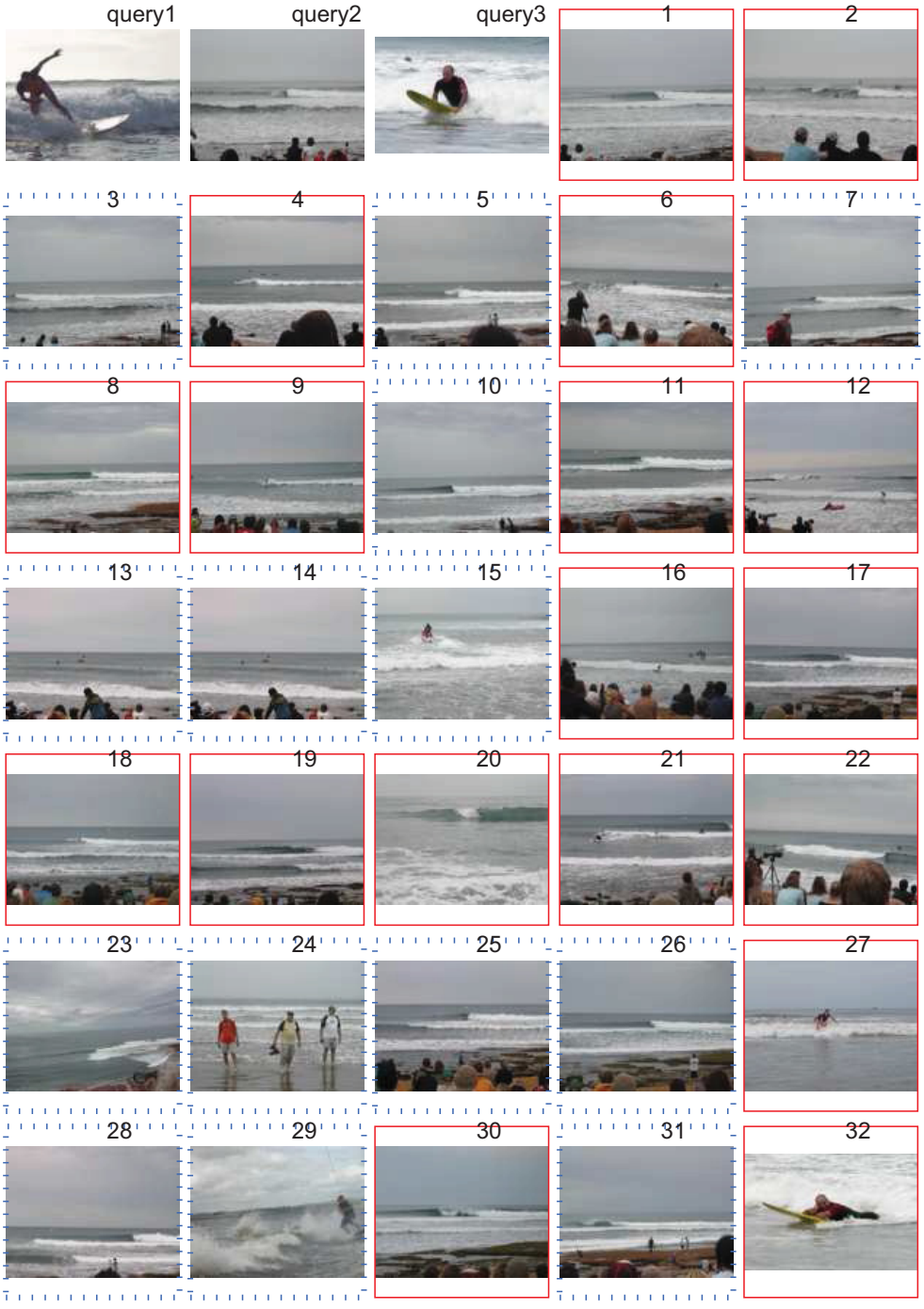

Figure 5.14: Retrieved images for the topic *people on surf boards* for the IAPR TC-12 data set. (Red rectangle: positive; blue dotted rectangle: negative)

Figure 5.15: Retrieved images for the topic *sunset over water* for the IAPR TC-12 data set. (Red rectangle: positive; blue dotted rectangle: negative)

to the query images with the sea, but not always having people on surf boards. Those images without people and surf boards are negative. In Figure 5.15, the retrieved images are all with the sun inside. But the topic requires images of sunset over water. Thus the images with sun, but not over water, or not with sunset are negative.

From these three examples, we can see that the retrieved target images by our methods show highly resemblance to the query images, but not always match the topic description. As a result, the precision results in Table 5.7 and 5.8 only provide the best MAP as 0.136. This result is partly caused by the semantic meaning in the topic, while the given query images only partially convey the semantics. The other reason is that the methods we apply to retrievals do not convey object level information or semantical meanings. The mismatching between the topic and image descriptors is known as the semantic gap [85]. Including methods that are able to describe objects in images can improve the retrieval performance to some extent, i.e., the SIFT features used by the XRCE group.

We also notice that the ground truth data given by the ImageCLEF may miss some positive images. For example, the target image ranking at 1 in Figure 5.13 is almost the same content as the second query image. But the ground truth data considers it as a negative match.

5.4 Summary

In this chapter, we compare the performance of our work in the thesis with the reported results by Deselaers et al. [19] for the Corel1K and UCID data sets. The performance is compared by two measures: the ER and the MAP. The comparison results show that our LTP histograms and RGB GMMs gain comparable results on both the data sets. The DCT2KL method outperforms other features for the Corel1K data set by a wide range, and gain the second best performance for the UCID dataset except the SIFT features. We further show the results by using our proposed features and methods, the LTP histograms, the RGB GMMs, and the DCT2KL method. The results show that the combination methods achieve better performance than using each of the methods separately. These features and methods represent one kind of color and texture joint distributions in different color spaces and domains, and thus the retrieval results are improved by combining them. We also show several retrieval

examples with different query images to provide visual impressions on how these features and methods affect the retrievals.

The methods are then applied to a large-scale image data set, i.e., IAPR TC-12 data set. This data set has 20,000 natural images, and has been used as the test set for the photographic retrieval tasks from 2006 to 2008. We show how our methods perform on the data set, and compared to those reported results also using visual features. From the results, we notice that to further improve the retrieval performance, local features should be included together to boost the performance by different information. Three MPEG-7 image descriptors are then added to our methods, and results show that the retrieval performance is improved to the second best of all reported results. Several retrieval examples on the data set show that our methods are able to find target images which are highly similar to the query images.

Chapter 6

Conclusion

6.1 Summary of Contributions

In this thesis, we present three methods for image matching in Content-Based Image Retrieval (CBIR). These three methods include: a method based on the LTP histograms; the second method based on the GMMs estimated by the EMass algorithm; and the third is called the DCT2KL method.

The LTP contains spatial information among neighboring pixels in an image. The spatial information provides more discriminative capability than the color levels. The LTP transforms the relationships of 9 neighboring pixels into a code, and the distribution of LTPs (histogram) in an image is used as the feature. Similarity between the LTP histograms is measured by the KL divergence. Compared to the related image features which also contain spatial information, the proposed LTP histograms achieve good discrimination capability and higher retrieval precision. We also suggest extracting the LTP histograms from a quantized color space and at different pattern length according to the application needs.

GMMs are alternate representation methods to histograms, when histograms are not suitable for describing the distribution of data, such as for high-dimensional data. GMMs give a compact and efficient representation to the high-dimensional data in images. The GMMs estimated by the EM algorithm usually suffer the local maxima problem. We present the EMass algorithm to estimate GMMs and avoid the local maxima by using the deterministic annealing method. The EMass algorithm estimates GMMs with high log-likelihoods more stable than the EM algorithm and the related DAEM algorithm. By applying the GMMs to represent the RGB color distributions in images, the GMMs estimated by the EMass algorithm also achieve higher precision than the other two algorithms.

The DCT2KL method addresses the representation of high-dimensional data through different methods. Motivated by image compression algorithms, we make use of image

processing techniques in different color spaces. More specifically, the DCT coefficients in the YCbCr color space are used instead of the color levels in the RGB color space. The DCT coefficients are restored by partially decoding JPEG images. Assume that each DCT coefficient sequence is emitted from a memoryless source, and all these sources are independent of each other. For each target image in the data set we form a hypothesis that its DCT coefficient sequences are emitted from the same sources as the corresponding sequences in the query image. Testing these hypotheses by measuring the log-likelihoods leads to a simple yet efficient scheme that ranks each target image according to the KL divergence between the empirical distribution of the DCT coefficient sequences in the query image and that in the target image. The DCT2KL scheme thus encompasses the image indexing and comparison together in a solid way.

Finally, we present a scheme to combine different features and methods to boost the performance of image matching. Detailed experimental results for each method on different image datasets, which include the Corel1K, the UCID, and the IAPR TC-12 image data sets, are provided. The performances of our methods are compared to the related works, and we give extensive discussion on their advantages and disadvantages. Examples showing retrieved images for different query images on different data sets are also provided to demonstrate the effectiveness of our methods and the combination scheme.

As a summary, the proposed work in our thesis is mainly focused on color and texture information. The LTP histograms provide local texture information from the spatial relations among neighboring pixels. The RGB GMMs provide the color distributions in the RGB color space. The DCT2KL method includes both the color information in the YCbCr color space by the DC coefficients, and the texture information by the AC coefficients. With the proposed methods, images are matched by using the color and texture information.

6.2 Future Work

Our work in the thesis shows that the global image features provide good retrieval results. However, especially on large-scale image datasets, different image features and methods should be combined to provide better retrieval results. This combination idea consists of two important directions for our future work. Firstly, what and

how to get more discriminating features for image matching. The retrieval results for the IAPR TC-12 dataset indicate that global features should be combined with other information to provide better retrieval results. The image local features, such as the SIFT features, are able to provide local object level information in images. We show initial experimental results by combining with localized MPEG-7 descriptors. The combination improves the retrieval precision. Thus, to further improve the retrieval performance, we plan to include more local features. As many local features are designed for object recognition purposes, these features are not able to apply to image matching for general image retrieval directly. Some efforts have been proposed, including Bag-of-Visual-Word [104], Model-based representation, clustering-based methods. However, each of these methods requires large quantity of training sets, as well as more work to improve the efficiency before combining with global features. Secondly, how to combine different features and methods in a more effective way? In this thesis, we present a combining scheme to equally sum the normalized similarities by different methods. We will work on tuning the weighting scheme, and the normalization for different features and methods.

We also notice another problem from the retrieval results for the IAPR TC-12 dataset. The features fail to represent images semantically, which is also known as the semantic gap. The semantic gap is defined in the survey by Smeulders et al. [85] as:

"the lack of coincidence between the information that one can extract from the visual data and the interpretation that the same data have for a user in a given situation."

How to bridge the semantic gap in image retrieval is very important for high level image retrieval. Despite current methods including relevance feedback techniques and building association between visual features and annotations, the bridging remains a major challenge for CBIR. We will work on how to bring semantics into our methods.

Bibliography

[1] Timo Ahonen and Matti Pietikäinen. Soft histograms for local binary patterns. *Proc. Finnish Signal Processing Symposium*, 2007.

[2] Jeffrey R. Bach, Charles Fuller, Amarnath Gupta, Arun Hampapur, Bradley Horowitz, Rich Humphrey, Ramesh C. Jain, and Chiao F. Shu. Virage image search engine: an open framework for image management. volume 2670, pages 76–87, 1996.

[3] Ricardo A. Baeza Yates and Berthier R. Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[4] Kobus Barnard, Pinar Duygulu, David Forsyth, Nando de Freitas, David M. Blei, and Michael I. Jordan. Matching words and pictures. *J. Mach. Learn. Res.*, 3:1107–1135, 2003.

[5] Muhammet Bastan, Hayati Cam, Ugur Gudukbay, and Ozgur Ulusoy. An MPEG-7 compatible video retrieval system with integrated support for complex multimodal queries. *Bilkent University, Technical Report (BU-CE-0905)*, 2009.

[6] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded up robust features. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision ECCV 2006*, volume 3951 of *Lecture Notes in Computer Science*, chapter 32, pages 404–417. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[7] Phil Brodatz. *Textures–A Photographic Album for Artists and Designers*. Dover, New York, 1966.

[8] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6):679–698, November 1986.

[9] Shi-Kuo Chang and Arding Hsu. Image information systems: Where do we go from here? *IEEE Transactions on Knowledge and Data Engineering*, 4:431–442, 1992.

[10] Yixin Chen and James Z. Wang. A region-based fuzzy feature matching approach to content-based image retrieval. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(9):1252–1267, 2002.

[11] Yixin Chen, James Z. Wang, and Robert Krovetz. CLUE: cluster-based retrieval of images by unsupervised learning. *Image Processing, IEEE Transactions on*, 14(8):1187–1201, 2005.

[12] Leszek Cieplinski. MPEG-7 color descriptors and their applications. In *Computer Analysis of Images and Patterns*, volume 2124, pages 11–20, New York, NY, USA, 2001.

[13] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, New York, NY, USA, 1991.

[14] Sagarmay Deb and Yanchun Zhang. An overview of Content-based Image Retrieval techniques. In *Proceedings of the 18th International Conference on Advanced Information Networking and Applications*, page 59, Washington, DC, USA, 2004.

[15] Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[16] Thomas Deselaers, Daniel Keysers, and Hermann Ney. Features for image retrieval: A quantitative comparison. *Pattern Recognition*, pages 228–236, 2004.

[17] Thomas Deselaers, Daniel Keysers, and Hermann Ney. FIRE - flexible image retrieval engine: ImageCLEF 2004 evaluation. *LNCS 3491, CLEF 2004*, pages 688–698, 2004.

[18] Thomas Deselaers, Daniel Keysers, and Hermann Ney. Discriminative training for object recognition using image patches. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 157–162, 2005.

[19] Thomas Deselaers, Daniel Keysers, and Hermann Ney. Features for image retrieval: an experimental comparison. *Information Retrieval*, 11(2):77–107, April 2008.

[20] James Dowe. Content-based retrieval in multimedia imaging. *Proceedings of SPIE Storage and Retrieval for image and video database*, 1993.

[21] Andrzej Drygajlo, Weifeng Lii, and Kewei Zhu. Verification of aging faces using local ternary patterns and q-stack classifier. In *Biometric ID Management and Multimodal Communication*, volume 5707 of *LNCS*, pages 25–32. Springer, 2009.

[22] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2 edition, November 2000.

[23] Guo-Can Feng and Jian-Min Jiang. JPEG compressed image retrieval via statistical features. *Pattern Recognition*, 36:977–985, 2003.

[24] Oscar Firschein, James Z. Wang, Gio Wiederhold, and ShaXin Wei. Content-based image indexing and searching using daubechies' wavelets. *International Journal on Digital Libraries*, 1(3):311–328, March 1998.

[25] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Qian Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video content: the QBIC system. *Computer*, 28(9):23–32, 1995.

[26] Andre Folkers and Hanan Samet. Content-based image retrieval using fourier descriptors on a logo database. In *ICPR '02: Proceedings of the 16 th International Conference on Pattern Recognition (ICPR'02) Volume 3*, pages 521–524, Washington, DC, USA, 2002.

[27] Arif Ghafoor. Multimedia database management systems. *ACM Comput. Surv.*, 27(4):593–598, 1995.

[28] Jacob Goldberger, Shiri Gordon, and Hayit Greenspan. An efficient image similarity measure based on approximations of KL-divergence between two gaussian mixtures. In *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*, pages 487–493, Washington, DC, USA, 2003.

[29] Jacob Goldberger, Shiri Gordon, and Hayit Greenspan. Unsupervised image-set clustering using an information theoretic framework. *IEEE Transactions on Image Processing*, 15(2):449–458, 2006.

[30] G. H. Granlund. Fourier preprocessing for hand print character recognition. *IEEE Trans. Comput.*, 21(2):195–201, 1972.

[31] Robert M. Gray. *Probability, Random Processes, and Ergodic Properties*. Springer, January 1990.

[32] Robert M. Gray and Richard A. Olshen. Vector quantization and density estimation. In *SEQUENCES '97: Proceedings of the Compression and Complexity of Sequences 1997*, pages 172–193, Washington, DC, USA, 1997.

[33] Michael Grubinger, Paul Clough, Allan Hanbury, and Henning Müller. Overview of the ImageCLEF 2007 photographic retrieval task. In *Working Notes of the 2007 CLEF Workshop*, Budapest, Hungary, 2007.

[34] Michael Grubinger, Paul Clough, and Clement Leung. The IAPR TC-12 benchmark for visual information search. *IAPR Newsletter*, 28(2):10–12, 2006.

[35] C. Harris and M. Stephens. A combined corner and edge detection. In *Proceedings of The Fourth Alvey Vision Conference*, pages 147–151, 1988.

[36] Chris Harris and Mike Stephens. A combined corner and edge detection. In *Proceedings of The Fourth Alvey Vision Conference*, pages 147–151, 1988.

[37] Daan He and Nick Cercone. Local Triplet Pattern for content-based image retrieval. In *ICIAR '09: Proceedings of the 6th International Conference on Image Analysis and Recognition*, pages 229–238. Springer-Verlag, 2009.

[38] Daan He, Nick Cercone, and Zhenmei Gu. Applying the extended mass-constraint EM algorithm to image retrieval. *Comput. Math. Appl.*, 56(4):1010–1024, 2008.

[39] Daan He, Zhenmei Gu, and Nick Cercone. Efficient image retrieval in DCT domain by hypothesis testing. *16th International Conference on Image Processing*, pages 225–228, 2009.

[40] Jing Huang, S. Ravi Kumar, Mandar Mitra, Wei-Jing Zhu, and Ramin Zabih. Image indexing using color correlograms. In *CVPR '97: Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, pages 762–768, Washington, DC, USA, 1997.

[41] Jesper Højvang Jensen, Daniel P.W. Ellis, Mads G. Christensen, and Søren Holdt Jensen. Evaluation of distance measures between mixture models of mfccs. *International Conference on Music Information Retrieval*, 2007.

[42] Jechang Jeong. The JPEG standard. pages 91–99, 1997.

[43] Sangoh Jeong and Robert M. Gray. A comparison of EM and GMVQ estimating gaussian mixtures: application to probabilistic image retrieval. *Acoustics, Speech, and Signal Processing (ICASSP 2005)*, 5(18–23):361–364, 2005.

[44] Sangoh Jeong, Chee Sun Won, and Robert M. Gray. Image retrieval using color histograms generated by Gaussian mixture vector quantization. *Comput. Vis. Image Underst.*, 94(1-3):44–66, 2004.

[45] Yan Ke and Rahul Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 2:506–513, 2004.

[46] M. L. Kherfi, D. Ziou, and A. Bernardi. Image retrieval from the World Wide Web: Issues, techniques, and systems. *ACM Comput. Surv.*, 36(1):35–67, March 2004.

[47] Gerald Kowalski and Mark T. Maybury. *Information Storage and Retrieval Systems: Theory and Implementation*. Kluwer Academic Publishers, Norwell, MA, USA, 2000.

[48] Iivari Kunttu, Leena Lepistö, and Ari Visa. Image correlogram in image database indexing and retrieval. *Proceedings of 4th European Workshop on Image Analysis for Multimedia Interactive Services*, pages 88–91, 2003.

[49] Jose A. Lay and Ling Guan. Image retrieval based on energy histograms of the low frequency dct coefficients. In *ICASSP '99: Proceedings of the Acoustics, Speech, and Signal Processing, 1999. on 1999 IEEE International Conference*, pages 3009–3012, Washington, DC, USA, 1999.

[50] Zaveta Levina and Peter Bickel. The Earth Movers Distance is the mallows distance: Some insights from statistics. *Proceedings of ICCV 2001*, 2:251–256, 2001.

[51] Michael Lew, Kim Lempinen, and Nies Huijmans. Webcrawling using sketches. *Proceedings of the 2nd International Conference on Visual Information Systems*, pages 77–84, 1997.

[52] Jia Li and James Z. Wang. Automatic linguistic indexing of pictures by a statistical modeling approach. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(9):1075–1088, 2003.

[53] Sven Loncaric. A survey of shape analysis techniques. *Pattern Recognition*, 31:983–1001, 1998.

[54] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004.

[55] Guojun Lu and Atul Sajjanhar. Region-based shape representation and similarity measure suitable for content-based image retrieval. *Multimedia Syst.*, 7(2):165–174, 1999.

[56] Zhe-Ming Lu and Hans Burkhardt. Colour image retrieval based on dct-domain vector quantisation index histograms. *Electronics Letters*, 41(17):956–957, 2005.

[57] Zhe-Ming Lu, Su-Zhi Li, and Hans Burkhardt. A content-based image retrieval scheme in JPEG compressed domain. *International Journal Innovative Computation, Information and Control*, 2(2):831–839, 2006.

[58] Wei-Ying Ma and B. S. Manjunath. Netra: a toolbox for navigating large image databases. *Multimedia Systems, Springer-Verlag, Berlin, Germany*, 7(3):184–198, 1999.

[59] Craig Macdonald, Iadh Ounis, and Ian Soboroff. Overview of the TREC-2007 blog track. *The Sixteenth Text REtrieval Conference (TREC 2007) Proceedings*, 2007.

[60] B.S. Manjunath, Jens-R. Ohm, Vinod V. Vasudevan, and Akio Yamada. MPEG-7 color and texture descriptors. *IEEE Trans. Circuits Syst. Video Technol.*, 11:703–715, 2001.

[61] Krystian Mikolajczyk, Tinne Tuytelaars, Cordelia Schmid, Andrew Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. 65(1-2):43–72, 2005.

[62] Carlton W. Niblack, Ron Barber, Will Equitz, Myron D. Flickner, Eduardo H. Glasman, Dragutin Petkovic, Peter Yanker, Christos Faloutsos, and Gabriel Taubin. QBIC project: querying images by content, using color, texture, and

shape. *Storage and Retrieval for Image and Video Databases*, 1908(1):173–187, 1993.

[63] Virginia Ogle and Michael Stonebraker. Chabot: Retrieval from a relational database of images. *IEEE Computer*, 28(9):40–48, Sep 1995.

[64] Timo Ojala, Topi Mäenpää, Matti Pietikäinen, Jaakko Viertola, Juha Kyllönen, and Sami Huovinen. Outex - new framework for empirical evaluation of texture analysis algorithms. *Proc. 16th International Conference on Pattern Recognition*, 1:701–706, 2002.

[65] Timo Ojala, Matti Pietikäinen, and Topi Mäenpää. Multiresolution gray scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002.

[66] Michael Ortega, Sharad Mehrotra, Kaushik Chakrabartia, and Kriengkrai Porkaewa. WebMars: A multimedia search engine. *Proceedings of the SPIE Electronic Imaging 2000: Internet Image*, 3964:314–321, 2000.

[67] Michael Ortega, Yong Rui, Kaushik Chakrabarti, Sharad Mehrotra, and Thomas S. Huang. Supporting similarity queries in MARS. In *MULTIMEDIA '97: Proceedings of the fifth ACM international conference on Multimedia*, pages 403–413, New York, NY, USA, 1997.

[68] Mira Park, Jesse S. Jin, and Laurence S. Wilson. Fast content-based image retrieval using quasi-gabor filter and reduction of image feature. *Southwest Symposium on Image Analysis and Interpretation*, pages 178–182, 2002.

[69] Greg Pass and Ramin Zabih. Histogram refinement for content-based image retrieval. *IEEE Workshop on Applications of Computer Vision*, pages 96–102, 1996.

[70] Greg Pass and Ramin Zabih. Comparing images using joint histograms. *ACM Journal of Multimedia Systems*, 7(3):234–240, May 1999.

[71] Alex Pentland, Rosalind W. Picard, and Stanley Sclaroff. Photobook: Content-based manipulation of image database. *International Journal of Computer Vision*, 18(3):233–254, Jun 1996.

[72] Florent Perronin and Chris Dance. Fisher kernels on visual vocabularies for image categorization. In *CVPR 2007*, pages 18–23, 2007.

[73] M. Pietikäinen, T. Mäenpää, and J. Viertola. Color texture classification with color histograms and local binary patterns. *Proc. 2nd International Workshop on Texture Analysis and Synthesis*, pages 109–112, June 2002.

[74] J. Puzicha, Y. Rubner, C. Tomasi, and J. Buhmann. Empirical evaluation of dissimilarity measures for color and texture. *International Conference on Computer Vision*, 2:1165–1173, 1999.

[75] Richard M. Rickman and T. John Stonham. Content-based image retrieval using color tuple histograms. *SPIE proceedings*, 2670:2–7, 1996.

[76] Kenneth Rose. Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. *Proc. of the IEEE*, 86(11):2210–2239, 1998.

[77] A. Sajjanhar, G. Lu, and J. Wright. An experimental study of moment invariants and fourier descriptors for shape based image retrieval. In *Proceedings of the Second Australia Document computing Symposium*, pages 46–54, April 5 1997.

[78] Atul Sajjanhar and Guojun Lu. A grid based shape indexing and retrieval method. *Computer Journal on Multimedia Storage and Archiving Systems*, 29:131–140, 1997.

[79] Phillipe Salembier and Thomas Sikora. *Introduction to MPEG-7: Multimedia Content Description Interface*. John Wiley & Sons, Inc., New York, NY, USA, 2002.

[80] Gerald Schaefer, Simon Lieutaud, and Guoping Qiu. CVPIC image retrieval based on block colour co-occurance matrix and pattern histogram. *International Conference on Image Processing*, pages 413–416, 2004.

[81] Gerald Schaefer and Michal Stich. UCID - an uncompressed colour image database. *Proc. SPIE, Storage and Retrieval Methods and Applications for Multimedia 2004*, pages 472–480, 2004.

[82] Stan Sclaroff, Leonid Taycher, and Marco La Cascia. ImageRover: A content-based image browser for the world wide web. *Proceedings of the IEEE Workshop on Content-Based Access of Image and Video Libraries*, pages 2–9, 1997.

[83] Michael Shenier and Mohamed Abdel-Mottaleb. Exploiting the JPEG compression scheme for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):849–853, August 1996.

[84] Amit Singhal and Google Inc. Modern information retrieval: a brief overview. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 24, 2001.

[85] Arnold W. M. Smeulders, Marcel Worring, Simone Santini, Amarnath Gupta, and Ramesh Jain. Content-based image retrieval at the end of the early years. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(12):1349–1380, 2000.

[86] Guy Smith and Ian Burns. Measuring texture classification algorithms. *Pattern Recognition Letters*, 18(14):1495–1501, 1997.

[87] John R. Smith and Shih-Fu Chang. Tools and techniques for color image retrieval. *SPIE proceedings*, 2670:1630–1639, 1996.

[88] John R. Smith and Shih-Fu Chang. Querying by color regions using the visualseek content-based visual query system. *Intelligent Multimedia Information Retrieval*, 7(3):23–41, 1997.

[89] David M. Squire, Wolfgan Müller, Henning Müller, and Julali Raki. Content-based query of image databases, inspirations from text retrieval: Inverted files, frequency-based weights and relevance feedback. *Scandinavian Conference on Image Analysis*, pages 143–149, 1999.

[90] Renato O. Stehling, Mario A. Nascimento, and Alexandre X. Falc ao. A compact and efficient image retrieval approach based on border/interior pixel classification. In *CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management*, pages 102–109, New York, NY, USA, 2002.

[91] Markus A. Stricker and Alexander Dimai. Color indexing with weak spatial constraints. *SPIE proceedings*, 2670:29–40, 1996.

[92] Markus A. Stricker and Markus Orengo. Similarity of color images. *SPIE Storage and Retrieval for Image and Video Databases*, 1995.

[93] Michael J. Swain and Dana H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(2):11–32, November 1991.

[94] Michael J. Swain, Charles Frankel, and Vassilis Athitsos. Webseer: An image search engine for the world wide web. *Technical Report TR-96-14, department of computer Science, University of Chichago*, 1996.

[95] Hideyuki Tamura, Shunji Mori, and Takashi Yamawaki. Texture features corresponding to visual perception. *IEEE Transactions on Systems, Man and Cybernetics*, 8(6):460–473, 1978.

[96] Xiaoyang Tan and Bill Triggs. Enhanced local texture feature sets for face recognition under difficult lighting conditions. In *Analysis and Modelling of Faces and Gestures*, volume 4778 of *LNCS*, pages 168–182. Springer, oct 2007.

[97] Naonori Ueda and Ryohei Nakano. Deterministic annealing EM algorithm. *Neural Networks*, 11:271–282, 1998.

[98] Nuno Vasconcelos. Minimum probability of error image retrieval. *IEEE Transactions on Signal Processing*, 52(8):2322–2336, 2004.

[99] Nuno Vasconcelos. On the efficient evaluation of probabilistic similarity functions for image retrieval. *IEEE Transactions on Information Theory*, 50(7):1482–1496, 2004.

[100] Remco C. Veltkamp and Mirela Tanase. Content-based image retrieval systems: A survey. *http://give-lab.cs.uu.nl/cbirsurvey/cbir-survey/index.html, a revised and extended version of Technical Report UU-CS-2000-34*, 2001.

[101] K. Wall. Curve fitting based on polygonal approximation. In *ICPR86*, pages 1273–1275, 1986.

[102] James Z. Wang, Jia Li, and Gio Wiederhold. SIMPLIcity: Semantics-sensitive integrated matching for picture libraries. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(9):947–963, 2001.

[103] James Z. Wang, Gio Wiederhold, Oscar firschein, and Shaxin Wei. Wavelet-based image indexing techniques with partial sketch retrieval capability. *Proceedings of the Fourth Forum on Research and Technology Advances in Digital Libraries*, pages 13–24, 1997.

[104] Jun Yang, Yu-Gang Jiang, Alexander G. Hauptmann, and Chong-Wah Ngo. Evaluating bag-of-visual-words representations in scene classification. In *MIR '07: Proceedings of the international workshop on Workshop on multimedia information retrieval*, pages 197–206, New York, NY, USA, 2007.

[105] C.H. Yeh and C.J. Kuo. Index-based fast search algorithm of image database on internet. *IEEE International Conference on Multimedia and Expo, 2000. ICME 2000*, 2:1195–1198, 2000.

[106] Atsuo Yoshitaka and Tadao Ichikawa. A survey on content-based retrieval for multimedia databases. *IEEE Transactions on Knowledge and Data Engineering*, 11(1):81–93, 1999.

[107] Hong Heather Yu. Visual image retrieval on compressed domain with Q-distance. *Computational Intelligence and Multimedia Applications*, pages 285–289, 1999.

[108] DaiDi Zhong and Irek Defée. Study of image retrieval based on feature vectors in compressed domain. *Proceedings of the 7th Nordic Signal Processing Symposium, 2006. NORSIG 2006.*, pages 202–205, 2006.

[109] DaiDi Zhong and Irek Defée. Image retrieval using efficient feature vectors generated from compressed domain. *15th European Signal Processing Conference (EUSIPCO 2007)*, pages 1580–1584, 2007.