Feature Extraction and Selection for Image Retrieval

Xiang Sean Zhou, Ira Cohen, Qi Tian, Thomas S. Huang
Beckman Institute for Advanced Science and Technology
University of Illinois at Urbana Champaign
Urbana, IL 61801
{xzhou2, iracohen, qitian, huang}@ifp.uiuc.edu

ABSTRACT

In this paper feature extraction process is analyzed and a new set of edge features is proposed. A revised edge-based structural feature extraction approach is introduced. A principle feature selection algorithm is also proposed for new feature analysis and feature selection. The results of the PFA is tested and compared to the original feature set, random selections, as well as those from Principle Component Analysis and multivariate linear discriminant analysis. The experiments showed that the proposed features perform better than wavelet moment for image retrieval in a real world image database; and the feature selected by the proposed algorithm yields comparable results to original feature set, and better results than random sets.

Keywords

Feature extraction, feature selection, content-based image retrieval, principle component analysis, discriminant analysis.

1. INTRODUCTION

Image retrieval problem, in some cases, can be regarded as a pattern classification problem, where each image is assumed—as ground truth—belongs to a specific class. Then query-by-example is to find the class and return images within that class. But usually there is no class labels available so this can only be achieved by comparing the "similarity" among images. Applications include medical image database retrieval, aerial/satellite image analysis and retrieval, etc., where the user is interested in some pre-defined targets such as tumors or vehicles. However in other cases, the assumption of fixed, static class membership assignments does not hold. One example is the trademark database retrieval. Even though trademarks are categorized (by hand!) according to their primary component, a new trademark design will be rejected if it is believed to cause confusion with an existing one, regardless of the category assignments. Another example can be illustrated by the image in Figure 1: as a query image submitted by the user, it is possible that the user is looking for "cars", or the user is looking for "autumn", or "house", etc. In general this is the case for every image in the database, i.e., each image has multiple "possible labels", or class memberships. And to complicate the issue further, different users can interpret the same image differently thus assign different labels in their mind, making training or pre-classification much more difficult. We refer the retrieval process in this case as dynamic matching.

Figure 1 An image of multiple objects

In this paper we will discuss the feature extraction and selection process with the aforementioned characteristics of image retrieval problem in mind. Especially for feature selection, which is defined as the process of selecting a subset from a set of original features, different cases will lead to different inputs and algorithms, which in turn will give different subsets of features. Without class labels, Principle Component Analysis (PCA), or a novel feature selection algorithm, Principle Feature Analysis (PFA), can be applied to transform the features to a reduced space, or to select the subset of the features that contains the most information, respectively. PFA is introduced in section 3.

For the image classification case, when some class labels are available, multivariate linear discriminant analysis (MLDA) is applied to transform the features into the Most Discriminating Feature space. The experimental results are compared in Section 4. For dynamic matching problem, contribution from each feature component would vary from time to time if discriminant analysis was conducted for different labeling scheme for the images. So the best way would be using dynamic on-line weighting of the features through relevance feedback [11]. If a reduced feature space is desired, it can only be done based on the results of sufficient trials of the relevance feedback process, so that a feature is deleted or ignored to a great extent only if it does not contribute in all or most of the dynamic weighting schemes. (See Figure 3)

Feature extraction is the process of creating a representation for, or a transformation from the original data. There has been a philosophical difference in attacking the problem of effective and efficient feature extraction. One is human perception-centered approach, i.e., based on human visual perception and

psychological experiments, compute the measurements of certain perception-based features (e.g., "smoothness" for texture), and then select the best mathematical representation for it. The other is machine-centered approach, in which case a unified computing scheme is selected for extraction of certain *ad hoc* features (e.g., "co-occurrence features", or "wavelet moments" for texture). The emphasis is first put on the computing aspect to provide an efficient algorithm to compute the "numbers"; and then the effectiveness, or the correlation between the "numbers" and human perception is established by experiments. Examples of the former include Tamura texture features [2], color quantization based on human perceptions, etc. A glance at the Appendix I of Haralick [1] will give the taste of the latter approach. Or, feature extraction from invisible bands for hyper-spectral image analysis, where there is no input to the human perception system at all. The perception-centered approach has the obvious advantages in serving the "master" or "user" of the image retrieval system—human, given that matching human performance is the ultimate goal of the system. But in other cases where human perception is not the target and performs not so well, while computer sees something human doesn't, the latter approach is more appropriate.

The two approaches were both used for color and texture feature extraction in the literature. But in terms of edge-based structure information, much less attention has been paid in both directions. (We define *structural features* as features *in-between texture and shape.* "Texture" captures spatial distribution of illuminance variations in the special form of "repeating patterns". "Shape" represents a specific edge feature that is related to object contour. Neither of them pays attention to information represented in non-repeating illuminance patterns, or more specifically, edge patterns in general, which is the category one can define the edge-based structural features to represent.) There have been attempts to use perception-based edge features for city/building image retrieval, i.e., use the features that human uses for recognizing buildings: edge direction; cotermination ("L" or "U" junctions) and parallel lines[7]; etc. We propose a machine-centered approach for general-purpose edge-based structural feature extraction. To illustrate the motivation, consider the edge maps in Figure 2. In all the three edge maps shape cannot be easily extracted. But a human subject can tell the top two as the Lena image and a building image. However this is not the case for the fox edge map at the bottom. But what can a computer do? Nothing much if "understanding" of the content (i.e., automatic segmentation and recognition of the object) is the goal. But if the goal is for image retrieval and matching only, i.e., find a representation which gives similar "numbers" whenever two images have similar content, and vice versa. Now the question becomes "how to effective and efficiently represent the information carried in an edge map?" In Section 2 we propose a revised version of our previous proposed water-filling algorithm[17] to extract more features from an edge map for the purpose of image matching and retrieval. (Note that this can work only to the extent where the invariance property required by the retrieval task is satisfied in the corresponding edge maps.) We also used the proposed Principle Feature Analysis algorithm to test whether the new features carry enough new information.

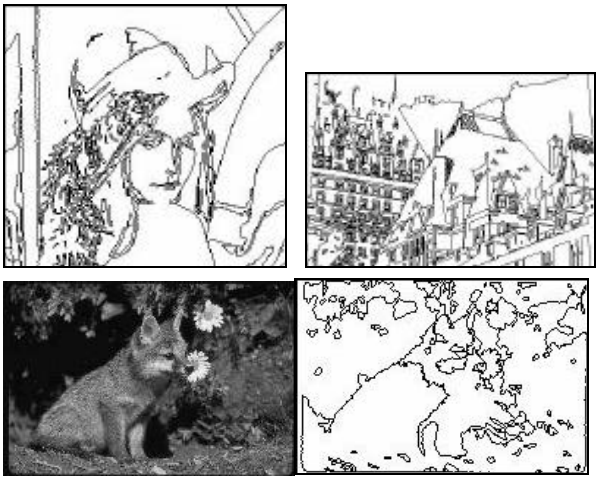

Figure 2 Edge maps

The scope of this paper is depicted in Figure 3, where the modules with a "*" are the emphasis of this paper.

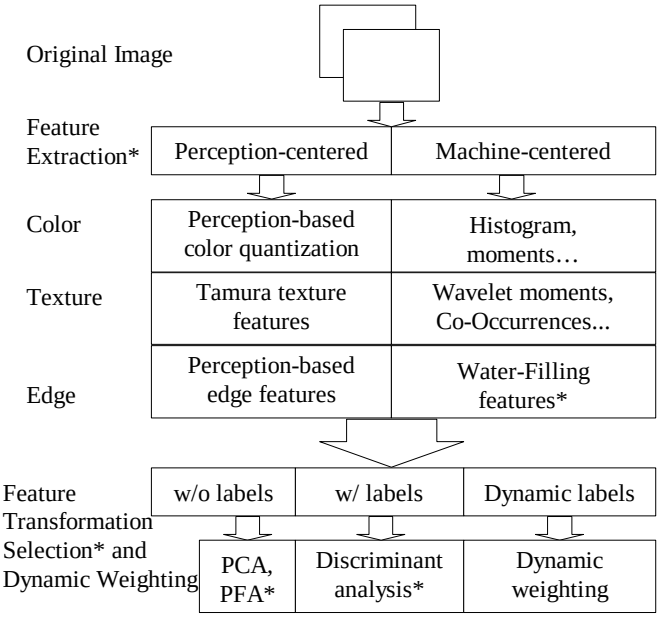

Figure 3 Scope of this paper

2. EDGE-BASED STRUCTURE FEATURE EXTRACTION

In this paper we use water-filling algorithm to extract edge/structural features. The advantages of this algorithm include efficiency—it is a linear-time algorithm; and effectiveness—multiple features corresponding to different human perceptions can be extracted simultaneously.

2.1 The Water-Filling Algorithm

We propose an algorithm to extract features from the edge map *directly* without edge linking or shape representation. The idea

is to look for measures for the *edge length* and *edge structure and complexity* by a very efficient graph traverse algorithm.

In this paper, we use 4-connectivity for simplicity. The algorithm also assumes that thinning operation has been performed on the edge map so that all the edges are one pixel wide.

To illustrate the algorithm, let's first consider the simple case of an 8 by 8 edge map with all the edge pixels connected (Fig. 4: shaded pixels are edge pixels). The algorithm will do a first raster scan on the edge map and start a traverse (think it as filling in water) at the first edge pixel encountered that has less than 2 neighbors, i.e., start at an end point. In Fig.4 the pixel with label "1" is the first end point encountered. (In case all edge pixels have at least two neighbors, i.e., no end points but all loops, e.g., Fig. 6, then start at the first unfilled edge pixel during the second scan. So it is necessary to have two raster scans to avoid possible miss). The waterfront then flows along the edges in the order indicated by the numbers. Note that when there are more than one possible paths to go at one point, the waterfront will fork (e.g., at pixel "6" and "9" in Fig. 1).

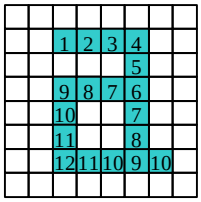

Figure 4

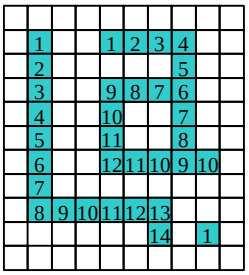

Figure 5

One can see that this algorithm can be regarded as a simulation of "flooding of connected canal systems (i.e., connected edges)", hence the name "water-filling algorithm". The assumptions implied by the algorithm include: i) we have unlimited water supply at the starting pixel; ii) water flows at a constant speed in all directions; iii) water front will stop only at a dead-end, or at a point where all possible directions have been filled.

When there are more than one set of connected edges in the edge map (e.g., the case in Fig.5), the algorithm will fill all the sets independently in sequential or in parallel.

As water fills the canals (edges), various information are extracted, which are stored as the feature primitives. Feature vectors can then be constructed based on these feature primitives. The time complexity of this algorithm is linear, proportional to the number of edge points in the image.

2.2 Feature Extraction

2.2.1 Feature Primitives

We propose the concept of "Feature primitives", which are defined as the quantities associated with or calculated from an image that can serve as bases for constructing feature vectors, often through using their statistics or entropies. Feature primitives can be used as feature vector directly as well, but often they are not compact enough. For example, co-occurrence matrices are the feature primitives for the co-occurrence texture features, most of which are moments, correlations, and entropies [1]; and wavelet transform coefficients can be regarded as feature primitives for wavelet based texture features such as wavelet moments. In our case, we propose the following quantities as structural feature primitives:

1). Filling time: Filling time is the time for water to fill a set of connected edges. For Fig. 4 through 6, the filling times are {12}, {14, 12, 1}, and {12}, respectively. Using different starting pixels, the filling time can vary in a range of [t, 2t], where t is the minimum filling time among all possible selection of the starting pixels. This is easily proved as follows: denote the starting pixel that gives the minimum filling time t as S. Since we assume water runs in both directions on an edge, water can reach S from any other starting pixel, say P, in time t, and then the waterfront can go from S to reach any pixels left unfilled within time t. So the filling time from P is =2t.

To minimize the variation in filling time due to selection of starting pixels, we can impose additional constraints on the selection of starting pixels (e.g., choose only the end points), or choose different starting pixels and average the results.

To achieve scaling invariance, normalize the filling time according to the image size. For example, divide filling time by (width + height).

2). Fork count: Fork count is the total number of branches the waterfront has forked during the filling of a set of edges. If we consider the initial waterfront as one branch to start with, then the fork count for Fig. 4 through 6 are {3}, {1, 3, 1}, and {6}, respectively. If we choose an end pixel as starting pixel whenever possible, fork count is "almost invariant" to starting pixel selection. The variation is within ±1, depend upon whether the water starts from the middle of an edge or the end of the edge. Also if multiple waterfronts collide at one intersection, even though the water does not actually fork, the potential forks should be counted to achieve the "almost invariance". E.g., an extra fork is counted both at the upper "9" and the lower "10" in Fig. 6; but none at "12', since it is not a potential fork point in any way).

3). Loop count: Loop count is the number of simple loops (or, "simple cycles" as defined in Corman et al., 1997, p. 88) in a set of connected edges. For example, in Fig.4 through 6, the loop counts are {1}, {0, 1, 0}, and {3}, respectively. Loop count is invariant to rotation.

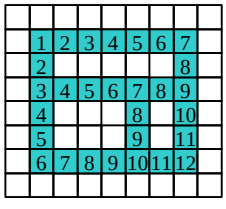

Figure 6

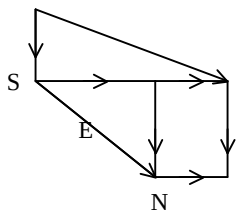

Figure 7

To get the loop count during the water-filling process, we make use of the following "Theorem of Splashes":

If we assume that when two waterfronts collide, we see one "splash"; or more generally, when n waterfronts collide at one intersection, we see n-1 "splashes" (think it as n-1 waterfronts collide with the first waterfront sequentially).

Then the number of splashes = the number of simple loops.

For example, in Fig. 6, three splashes are recorded at time "9", "10", and "12". Hence the loop count is 3.

Proof: Treat the set of connected edges as a graph; and regard the starting point, the fork points, and the collision/splash points as the nodes of the graph; and the water branches as the edges. For example, the corresponding graph for Fig. 6 is shown in Fig. 7. Then the water-filling process is a traverse of the graph, and the total number of splashes is the total number of revisits to some of the nodes, which is the number of simple cycles/loops in the graph.

The above theorem provides a way of recording loop counts within the water-filling process with very little overhead computation.

4). Water amount: Water amount is the total amount of water used to fill up the set of edges in terms of number of pixels. So it is the edge pixel count. In Fig. 4 through 6, the water amounts are {18}, {14, 18, 1}, and {29}, respectively.

5). Horizontal (vertical) cover: Horizontal (vertical) cover is the width (height) of the rectangular bounding box of the set of edges. In Fig. 4 through 6, the horizontal covers are {5}, {6, 5, 1}, and {7}, respectively.

6). Longest horizontal (vertical) flow: Longest horizontal (vertical) flow is the longest horizontal (vertical) edge in the set of connected edges. For Fig. 4 through 6, the longest vertical flows are {6}, {8, 6, 1}, and {6}, respectively.

Note that there exist many other possibilities on selecting the feature primitives. However, the final selection should depend upon the specific application, i.e., what information is important and most discriminative toward the classification.

2.2.2 Edge/structural feature formation

Based on the feature primitives in 4.1, we can then construct edge/structural features from their statistics. For example: moments (e.g., average filling time); order statistics (e.g., maximum loop count); distributions (e.g., water amount histogram); etc. In the following we discuss some examples with the emphasis on their meanings from a human perception point of view.

1). MaxFillingTime and the associated ForkCount

MaxFillingTime(MFC) is defined as max{filling times}. For Fig. 4 through 6, the MFC is 12, 14, and 12, respectively. And the associated ForkCount(FC) is 3, 1, and 6 respectively. So the MFT&FC vectors for Fig. 4 through 6 are (12, 3), (14, 1), and (12, 6), respectively.

MFT&FC are features most probably associated with a salient object (with the "longest" edge) in the image. The MFT conveys a rough measure of the size (edge length) of this object, while the associated FC gives measure of complexity of the structure of the object (complexity of the edges).

2). MaxForkCount and the associated FillingTime

Similarly defined as MFT&FC, these are also features most probably associated with a salient object in the image. The MFT conveys a rough measure of the complexity of the object. This object may or may NOT be the same object as the previous one. For Fig. 4 and 6, the MFC&FT is the same as the MFT&FC. But for Fig. 5, the MFC&FT vector is (3, 12).

3). (GLC&MLC) GlobalLoopCount and MaxLoopCount

GlobalLoopCount is defined as sum{loop counts}. MaxLoopCount is max{loop counts}. This feature vector can capture structural information such as the windows in the build images. Or can be used toward character detect and recognition applications.

4). (FTH&FC) FillingTime Histogram and the associated averaged ForkCount within each bin

This is a global feature on all sets of connected edges in the edge map. It represents the edge map by the distribution of edge "length". Noise or changing background with short edges may only affect part of the histogram, leaving the portion depicting the salient objects unchanged. Thus by proper weighting of the components (e.g. by relevance feedback[11]), we could achieve robust retrieval.

5). (WAH) WaterAmount Histogram

This is also a global feature with multiple components. It is another measure of the distribution in edge length or density.

Note again that there can be many other possible ways to construct feature vectors, such as the moments of filling times, fork counts, loop counts, or water amounts, etc.,

3. FEATURE TRANSFORMATION AND SELECTION

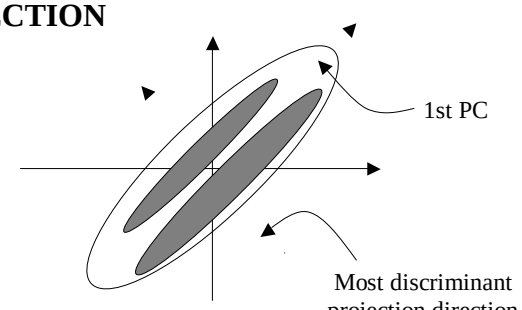

Figure 8

For any handcrafted feature set, data-dependent analysis can always be carried out to select an optimal projection or transformation to best represent the information carried in the original feature set, or reduce the dimensionality with minimum information loss. When there is no label for the data, principle components analysis [9] gives the best linear transform from the original feature set in terms of reconstruction errors. If class labels are (partially) available,

then multivariate linear discriminant analysis (MLDA) [15] gives the best linear transformation in terms of maximizing the ratio of inter-class scatter over in-class scatter, i.e., the MLDA vector has the most discriminating power among all linear transformations. The resulting transformation of PCA or MLDA is in general different. Figure 8 shows the data scatter of the whole dataset, and two classes by three eclipses, and the two projection directions are shown in dotted line. Both of the above techniques require the original feature set for the calculation of the transformed features. If computation is an issue in the feature extraction process, or some of the original feature component is of special meaning and the user want to keep them unchanged, while dimension reduction is still desired, then feature selection technique is needed. [18] gives a detailed comparison on the feature selection algorithms for labeled data. Here we propose a Principle Feature Selection algorithm for un-labeled data, i.e., this algorithm will output the subset of the features that will best represent original data.

3.1 Principle Component Analysis (PCA)

Principle components are the projection of the original features onto the eigenvectors corresponds to the largest eigenvalues of the covariance matrix of the original feature set. Principle components provide linear representation of the original data using the least number of components with the mean-squared error minimized. (See [9] for details)

3.2 Principle Feature Analysis (PFA)

Let X be a zero mean n-dimensional random feature vector. Let S be the covariance matrix of X (which could be in correlation form as well). Let A be a matrix whose columns are the orthonormal eigenvectors of the matrix S, computed using the singular value decomposition of the S:

$$\Sigma = A\Lambda A^T \tag{1}$$

where $\Lambda = \begin{bmatrix} l_1 & & \\ & \ddots & 0 \\ & 0 & \ddots \\ & & & l_n \end{bmatrix}$, $l_1 \geq l_2 \geq \ldots \geq l_n$

and $A^T A = I_n$

Let A_q be the first q columns of A and let $V_1, V, \ldots V_n \in \Re^q$ be the rows of the matrix A_q.

The vector V_i corresponds to the i'th feature (variable) in the vector X, and the coefficients of V_i correspond to the weights of that feature on each axes of the subspace. Features that are highly correlated will have similar absolute value weight vectors (absolute value is necessary since changing the sign of one variable changes the signs of the corresponding weights but has no statistical significance [9]). In order to find the best subset we will use the structure of these rows to first find the features which are highly correlated to each other and then choose from each group of correlated features the one which will represent that group optimally in terms of high spread in the lower dimension, reconstruction and insensitivity to noise. The algorithm can be summarized in the following five steps:

Step 1 Compute the Sample Covariance matrix, or use the true covariance matrix if it is available. In some cases it would be preferred to use the Correlation matrix instead of the Covariance matrix. The Correlation matrix is defined as the *nxn* matrix whose *i,j'th* entry is

$$r_{ij} = \frac{E\lfloor x_i x_j \rfloor}{E[x_i^2]E[x_j^2]}$$

This representation is preferred in cases where the features have very different variances from each other, and using the regular covariance form will cause the PCA to put very heavy weights on the features with the highest variances. See [9] for more details.

Step 2 Compute the Principal components and eigenvalues of the Covariance/Correlation matrix as defined in equation (1).

Step 3 Choose the subspace dimension q and construct the matrix A_q from A. This can be chosen by deciding how much of the variability of the data is desired to be retained. The retained variability can be computed using:

$$Variability\,\mathrm{Re}\,tained = \frac{\sum_{i=1}^{q} l_i}{\sum_{i=1}^{n} l_i} \cdot 100\%$$

Step 4 Cluster the vectors $|V_1|, |V_2|, \ldots, |V_n| \in \Re^q$ to $p \geq q$ clusters using K-Means algorithm. The distance measure used for the K-Means algorithm is the Euclidean distance. The vectors are clustered in p clusters and the means of each cluster is computed. This is an iterative stage which repeats itself until the p clusters are found and do not change. The reason to choose p greater then q in some cases is if the same retained variability as the PCA is desired, a slightly higher number of features is needed (Usually 1-5 more are enough).

Step 5 In each cluster, find the corresponding vector V_i which is closest to the mean of the cluster. Choose the corresponding feature x_i as a principal feature. This will yield the choice of p features. The reason for choosing the vector nearest to the mean is twofold. This feature can be thought of as the central feature of that cluster- the one most dominant in it, and which holds the least redundant information of features in other clusters. Thus it satisfies both of the properties we wanted to achieve- large "spread" in the lower dimensional space, and good representation of the original data. Note that if a cluster has just two components in it, the chosen feature will be the one

which has the highest variance of the coefficients, since this means that it has better separability power in the subspace.

By choosing the principal features using this algorithm, we choose the subset that represents well the entire feature set both in terms of retaining the variations in the feature space (by using the clustering procedure), and keep the prediction error at a minimum (by choosing the feature who's vector is closest to the mean of the cluster). (See [19] for more details.)

3.3 Multivariate Linear Discriminant Analysis (MLDA)

Assume that S_w is the in-class scatter matrix, S_b is the inter-class scatter matrix, then the most dicriminant projection directions are the eigenvectors of the $S_w^{-1}S_b$ associated with the largest eigenvalues. (For details see [15])

4. Experiments and Comparisons

The first set of experiments is designed to test the performance of the water-filling features versus the widely used wavelet-moments as texture measures. Water-filling features used in these experiments are MFT&FC (see Section 2 for definition), MFC&FT, GLC&MLC, and FTH&FC (7 bins)—a total of 18 feature components per image. Euclidean distance is used as distance measure. The datasets from Corel consists of 17695 images, 400 of which are "airplanes", and 100 of them are "American eagles". Table 1 shows the comparison in terms of averaged number of hits in top 10, 20, 40 and 80 returns for 100 airplanes and 100 eagles as the query images, respectively.

Table. Water-filling (WF) versus Wavelet Variances (WM)

Airplanes	**#Hit in top 10**	**Top 20**	**Top 40**	**Top 80**
WF	3.5600	6.2900	10.9200	18.0300
WV	3.3200	5.7500	9.9400	17.0700
Eagles	**#Hit in top 10**	**Top 20**	**Top 40**	**Top 80**
WF	2.6538	3.3269	4.9135	6.7885
WV	1.9808	2.8173	4.4327	6.5769

Principle feature analysis is applied on the joint feature set, which is the union of the Water-filling features and wavelet moment features. For three datasets, 2, 2, and 3 pairs of features from different sets are clustered together, all with low correlation coefficients (<0.5), which indicates that water-filling features contain mostly new information than that is already expressed by wavelet moments.

The second sets of experiments are designed to test the output of the principle feature analysis algorithm (PF) for the purpose of image retrieval. Table 2 shows the retrieval results on the same sets of query images as the first experiment (100 airplanes and 100 eagles). The fourth test in each case corresponds to the same number of features selected, but they span less number of clusters. One can see that the performance is consistently lower. Table 3 shows the testing results on the VISTEX texture database from MIT Media Lab. In this database the original images are tiled into 4X4=16 sub images for testing purpose, so there are 52 big images and 832 resulting sub images. Since we have the classes and labels, MLDA is tested to compare with all the other cases. One can clearly see that MLDA outperform all the cases. This is reasonable since it utilizes extra information. Please note that the last test case in Table 3 have more feature components (6) than that of the principle features selected (5), but since these feature components are chosen as clustered ones (feature 2, 4, 8 and 10 are all clustered into one cluster for this dataset!), so the performance is even worse, even though one might intuitively expect it to perform better since they spanned all the three wavelet sub-bands!

Table 2. Principle Features (PF) Performance Comparison using 10 wavelet variances (WV) for Corel dataset

Airplanes	**#Hit in top 10**	**Top 20**	**Top 40**	**Top 80**
10 WV's	3.3200	5.7500	9.9400	17.0700
7 PC's	3.3700	5.7100	9.9000	16.8000
7 PF's (1,5,6,7,8,9,10)	3.5800	6.2400	11.1500	19.2900
(1,2,3,4,8,9,10)	3.2000	5.3400	9.2500	15.5600
Eagles	**#Hit in top 10**	**Top 20**	**Top 40**	**Top 80**
10 WV's	1.9808	2.8173	4.4327	6.5769
7 PC's	1.9615	2.7596	4.1250	6.2692
7 PF's (1,5,6,7,8,9,10)	2.1442	2.9615	4.6731	7.2115
(1,2,3,4,8,9,10)	1.7788	2.5000	3.7115	5.7500

Table 3. Principle Features (PF) Performance Comparison using 10 wavelet variances (WV) for VISTEX dataset

VISTEX	**#Hit in top 10**	**Top 20**	**Top 40**	**Top 80**
10 WV's	8.2632	12.1562	13.7308	14.7175
5 PC's	8.0505	11.8197	13.5577	14.6370
5 MLDA's	8.3702	12.6755	14.2800	15.1671
5 PF's (1,5,6,7,10)	7.6575	11.3293	13.1659	14.4351
(2,3,4,6,8,10)	7.3329	10.7812	12.9087	14.254

Table 4. Principle Features (PF) Performance Comparison using 9 color moments and 20 wavelet moments (WM)

40 random queries in Corel	**#Hit in top 10**	**Top 20**
All:9 color + 20 WM	4.98	8.23
22 PF's (7 color, 15 WM)	4.90	8.15

22 random selected (I) (7 color, 15 WM)	3.88	6.65
22 random selected (II) (7 color, 15 WM)	4.18	6.85
22 random selected (III) (7 color, 15 WM)	4.40	7.18

Table 4 shows results on the 9 color moments plus 20 wavelet moments on the Corel database. 7 out 9 color features and 15 out of 20 wavelet feature are selected by PFA. 40 randomly selected query images are picked. The performance of the original feature set and the principle set are tested against 3 randomly selected (but the same number of) features. It clearly shows that Principle features yield comparable results as that of original set and significantly higher results than any random picks.

5. CONCLUSIONS

In this paper we proposed a revised edge-based structural feature extraction approach by following guidelines obtained from summarizing the existing feature extraction approaches. A principle feature selection algorithm is also proposed for new feature analysis and feature selection. The results of the PFA is tested and compared to the original feature set, random selections, as well as those from Principle Component Analysis and multivariate linear discriminant analysis. The experiments showed that the proposed algorithm yield comparable results to original set, and better results than random sets.

6. REFERENCES

[1] Haralick, R. M., K. Shanmugam, and I. Dinstein, Texture feature for image classification, *IEEE Trans. System Man and Cybernetics* (Nov., 1973)

[2] Tamura, Hideyuki, S. Mori, and T. Yamawaki, Texture Features Corresponding to Visual Perception, *IEEE Trans. System Man and Cybernetics*, 8 (6), (1978).

[3] Corman, T. H., C. E. Leiserson, R. L. Rivest. *Introduction to algorithms*, McGraw-Hill, New York, 1997

[4] Flickner, M. et al., Query by image and video content: The qbic system, *IEEE Computers*, 1995

[5] Gonzalez, R. C. and Woods, *Digital Image Processing*, Addison-Wesley, 1992

[6] Hu, M. K., Visual pattern recognition by moment invariants", *IRE Trans. Information Theory*, 8, 1962

[7] Iqbal, Q. and J. K. Aggarwal, Applying perceptual grouping to content-based image retrieval: Building images, Proc. IEEE CVPR'99, 42-48, 1999

[8] Jain, A. K., *Fundamentals of Digital Image Processing*, Prentice Hall, 1989

[9] Jolliffe, I.T., *Principal Component Analysis*, Springer-Verlag, New-York, 1986.

[10] Laine, A., J. Fan. Texture classification by wavelet packet signatures. *IEEE Trans. Pattern Anal. Machine Intell*. 15, 1993

[11] Rui, Y., T. S. Huang, M. Ortega, and S. Mehrotra, Relevance Feedback: A Power Tool in Interactive Content-Based Image Retrieval", *IEEE Tran on Circuits and Systems for Video Technology*, Vol 8, No. 5, Sept., 644-655, 1998

[12] Smith, J. R. and Chang, Transform features for texture classification and discrimination in large image databases, *Proc. IEEE ICIP*, 1995

[13] Tabb, M., N. Ahuja, "Multiscale Image Segmentation by Integrated Edge and Region Detection", *IEEE trans. Image Processing*, Vol. 6, No. 5, May 1997

[14] Vasconcelos, N. and A. Lippman, Embedded mixture modeling for efficient probabilistic content-based indexing and retrieval, in *SPIE Multimedia Storage and Archiving Systems III*, Boston, 1998

[15] Wilks, S.S., *Mathematical Statistics*. New York: Wiley, (1963)

[16] Zahn, C. T. and Roskies, Fourier descriptors for plane closed curves, *IEEE Trans. Computers*, 1972

[17] Zhou, X. S., Y. Rui, and T. S. Huang, Water-filling: a novel way for image structural feature extraction, *Proc. ICIP*, 1999

[18] Jain, A., A.,Zongker, "Feature Selection: Evaluation, Application, and Small Sample Performance", *IEEE Trans. PAMI*, VOL. 19, (Feb, 1997)

[19] I. Cohen, Q. Tian, X. Zhou, T. S. Huang, "Feature Selection and Dimensionality Reduction Using Principle Feature Analysis", submitted to the Sixteenth Conference on Uncertainty in Artificial Intelligence (UAI-2000) Workshop on Fusion of Domain Knowledge with Data For Decision Support, Stanford University, Stanford, CA, June 30, 2000.