# MULTI-LEVEL DISCRETE COSINE TRANSFORM FOR CONTENT-BASED IMAGE RETRIEVAL BY SUPPORT VECTOR MACHINES

*Yong Li*, Xiujuan Chen, Xuezheng Fu, and Saeid Belkasim*

Department of Computer Science, Georgia State University, Atlanta, GA 30303
*Contact Author: cscyxlx@cs.gsu.edu

## ABSTRACT

Texture feature extraction is widely used in content-based image retrieval (CBIR) and is not efficient to be implemented directly in the pixel domain due to high information redundancy and strong correlations in raw images. It is well known that low-frequency coefficients of the Discrete Cosine Transforms (DCTs) preserve the most important image features. In this paper, we use Multi-level DCTs (MDCTs) to generate image texture feature vectors for the purpose of CBIR. The texture feature vectors generated from MDCTs coefficients and Zernike moments are classified by Support Vector Machines (SVMs). The experimental result shows good average retrieval accuracy. It also shows that DCT coefficients from low level resolution images are sufficient to extract image texture feature with significant less computing cost.

***Index Terms***— Multi-level Discrete Cosine Transform, CBIR, Feature Extraction, Zernike Moments, SVMs

## 1. INTRODUCTION

In image retrieval, image features such as texture, shape, spatial layout and color are used to specify queries. Texture contains important information about the structural arrangement of surfaces and their relationship to the surrounding environment [1]. Texture is the most important visual cue in identifying a variety of materials. It can be used in, for example, remote sensing to obtain the boundary map separating the differently textured regions, and image compression where synthesis texture may replace backgrounds in natural scenes thus leading to a dramatic bit saving [2].

A variety of techniques, ranging from statistical methods to multi-resolution filtering have been developed for texture analysis. Two-dimensional Gabor Filter as one of the multiresolution filtering techniques, is proved to be very useful in texture analysis and is widely adopted in the literature [3][4][5][6]. Manjunath and Ma [3] have shown that image retrieval using Gabor Filter outperforms that using various wavelets transform, including orthogonal and bi-orthogonal wavelet transform, and tree-structured wavelet transform. However, the computational cost of using the above wavelet transforms is expensive for image retrieval. Normally, the retrieval mechanisms make similarity measure by contrasting the features of the query image and the features from the images in the database. An efficient as well as simple feature extraction scheme is obligatory for real-time image retrieval.

It is well known that low-frequency coefficients of the Discrete Cosine Transforms (DCTs) preserve the most important image features. In this paper, we use Multiple-level DCT (MDCT) coefficients to create texture feature vectors. The texture feature vectors generated from MDCT coefficients and Zernike moments are trained and classified by Support Vector Machines (SVMs), which have demonstrated powerful and promising generalization abilities in image processing and other classification applications.

The rest of the paper is organized as follows. Section 2 discusses image texture features and feature extractions from MDCTs. Zernike moments are also introduced in this section. Section 3 discusses the basic concepts of SVMs and how to use them in our classification and retrieval schema. Experimental results and conclusions are outlined in Section 4.

## 2. TEXTURE FEATURE EXTRACTION AND MULTI-LEVEL DISCRETE COSINE TRANSFORM

The discrete cosine transform (DCT) is often used in signal and image processing, especially for lossy data compression. The excellent energy compaction property of the DCT is the main reason for its popularity. This property enables most of the signal information to be concentrated in a few low-frequency components of the DCT [7] [8]. Smith and Chang [9] compared several subband-energy features which can be used for texture classification. In [10], Huang introduced extracting texture features directly from the DCT coefficients in the DCT-code image.

### 2.1. DCT Coefficients as Image Texture Features

The process of image classification using DCT coefficients can be summarized as follows: An input image is partitioned into sub-blocks with the size of $N \times N$. DCT is performed on each block. There are total $N^2$ coefficients within each block and the variance and mean values of each coefficient among the blocks are calculated to generate $2N^2$ features. The $2N^2$ feature vectors generated by the training images are mapped into a reduced space using Fisher Discriminant Analysis, which works by finding the eigenvectors of scatter matrices. A subset of the resulting eigenvectors that account for the largest total variation is used to assign new texture image to the nearest classes.

### 2.1. Multi-level Discrete Cosine Transform

In this paper, we use MDCT coefficients which are derived from the same image with different resolutions and Zernike moments of the images to create image texture feature vectors. We define the 1-D Multi-level Discrete Cosine Transform as:

$$MDCT_L(k) = \sum_{i=0}^{N/2^L-1} 2^{-L} \left( \sum_{j=2^L*i}^{2^L*(i+1)-1} V(j) \right) \cos\left[ \frac{\pi * 2^L}{N} \left(j + \frac{1}{2}\right) k \right] \quad (1)$$

where $k \in [0, {}^{N}\!/_{2^L} - 1]$ for any level $L$, $L \in [0, \log_2 N\text{-}1]$.

From (1), we can see that MDCTs with different level *L* is the DCTs of the same signal with a different resolution. When *L* equals to 0, MDCT is the standard form of DCT-II. This formula can be extended to 2-D images since 2-D DCT can always be implemented by first applying a 1-D DCT on all rows, followed by a 1-D DCT on all columns from the result of the first step. To calculate MDCT of an image in Matlab, we can simple apply *dct2* function to the original image which has be been resized with the compression factor of $2^l$.

The multi-resolution DCTs presented here are derived in the hope that they can represent the image texture features from its low resolution such that the processing time will be significantly reduced. For example, the total computing cost of $MDCT_2$ and $MDCT_4$ is about $1/4$ and $1/16$ of standard DCT when applying to an image.

Secondly, in some cases, MDCT may detect the difference of the image texture features resided in two images while the standard DCT fails to do so. For example, suppose we want to distinguish the following two images shown in Fig. 1, both have a size of 64 by 64:

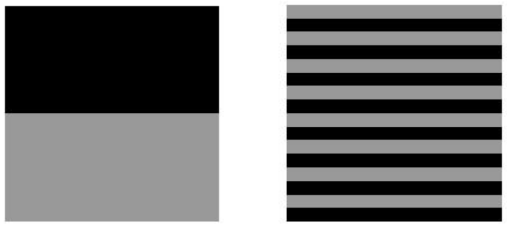

Fig. 1. (a) Square image, dark and gray even divided
(b) Stripe image, stripe size: 4 by 64

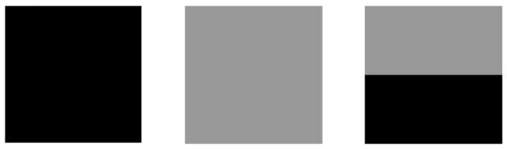

Fig. 2. Three 4 by 4 image blocks.

We divide both images into 4 by 4 blocks as mentioned in [9]. As a result, both images have two kinds of 4 by 4 blocks: pure dark and pure gray as shown in Fig. 2a and 2b. In this case, we do not have to go through calculating the DCT coefficients before we can tell that the feature vectors extracted from the two images are identical, since both contain the same 128 black blocks and 128 gray blocks. Alternatively, we can use $MDCT_2$ to fetch the feature vectors. As we discussed above, implementing $MDCT_2$ is same as implementing standard DCT-II on the low resolution images with the compression ratio of 2. The low resolution images are similar to the ones in Fig. 1 with the exceptions: both images have a reduced size of 32 by 32; stripe size in Fig. 1b decreases to 2 by 32. Since we still use 4 by 4 sub-bands, we can see, Fig. 1a is divided into dark/gray blocks and Fig. 1b is divided into blocks of half gray and half dark as shown in Fig. 2c. When we apply DCT to the blocks in Fig. 2a, 2b and 2c, we can expect all zero values of the AC coefficients in Fig. 2a and 2b since 'no-change' occurs in these two blocks. There must be non-zero AC coefficients in the block of Fig. 2c. Thus, the feature vectors extracted from Fig. 1a and 1b by using $MDCT_2$ will be different and be able to be used to distinguish one from the other.

### 2.2. Image feature from Zenike Moments

We also use Zenike Moments(ZM) to extract image features in this approach. Zernike moments have many desirable properties, such as rotation invariance, robustness to noise, expression efficiency. The complex ZM are derived from Zernike polynomials which are a set of complex, orthogonal polynomials defined over the interior of a unit circle $x^2 + y^2 = 1$.

$$V_{nm}(x, y) = V_{nm}(\rho, \theta) = R_{nm}(\rho) \exp(jm\theta) \quad (2)$$

$$R_{nm}(\rho) = \sum_{s=0}^{\frac{n-|m|}{2}} (-1)^s \frac{(n-s)!}{s!(\frac{n-|m|}{2}-s)!(\frac{n-|m|}{2}-s)!} \rho^{n-2s} \quad (3)$$

where n is a non-negative integer, m is an integer such that n-|m| is even and |m|≤n, $\rho = \sqrt{x^2 + y^2}$, and $\theta = \tan^{-1} \frac{y}{x}$

Projecting the image function onto the basis set, the Zernike moments of order *n* with repetition m is given by:

$$A_{nm} = \frac{n+1}{\pi} \sum_x \sum_y f(x,y) V_{nm}(x,y) \; , \; x^2 + y^2 \le 1 \quad (4)$$

It has been shown that the ZM on a rotated image have the same magnitudes. Therefore $|A_{nm}|$ can be used as a rotation invariant feature of the image function. It has also been shown in [11] that a ZM order of 4 or 6 are suitable for image feature description. In this paper 9 ZM moments are included in the feature vectors.

## 3. IMAGE FEATURE TRAINNING AND CLASSIFICATION USING SVM

SVMs are used to train the image feature vectors to classify different texture images. Given a testing image, SVMs will predict its texture class based on its texture feature vector. This process involves binary classification and mulit-class classification.

### 3.1. Binary Classification

Assume there is a training data set *S*: $\{(x_i, y_i)\}_{i=1}^N$, where each input $x_i \in \Re^m$ and output $y_i \in \{\pm 1\}$. The goal of SVMs is to find an optimal hyperplane $w \cdot z + b = 0$ in a *feature space*, which can be transformed from the input vector space $\boldsymbol{x}$ by mapping $z = \phi(x)$, and separate the training data into two classes with the maximum margin in the feature space, where $w = \sum_{i=1}^N \alpha_i y_i z_i$, $\alpha_i$ is a set of Lagrange multipliers to the following dual problem [12]:

$$\textit{Maximize}: \; W(\alpha) = \sum_{i=1}^N \alpha_i - \tfrac{1}{2} \sum_{i,j=1}^N \alpha_i \alpha_j y_i y_j (z_i \cdot z_j)$$

$$\textit{Subject to}: \; C \ge \alpha_i \ge 0, \sum_{i=1}^N \alpha_i y_i = 0 \, . \quad (5)$$

where $C$ is a user-defined regularization parameter, determining the tradeoff between maximizing margin and minimizing the number of training examples misclassified. It is useful to handle non-separable problems and outliers.
The examples $\mathbf{x}_i$ with $\alpha_i > 0$ are called support vectors. They lie close to the decision boundary. If support vectors are removed, the separating hyperplane would be changed.
The *kernel trick* of SVMs allows us to substitute the dot product of data points in (2) with just a *kernel function*:

$$K(x_i \cdot x_j) = z_i \cdot z_j \quad (6)$$

The decision function is made by computing

$$f(x) = sign(w \cdot z + b) = sign(\sum_{i=1}^N \alpha_i y_i K(x_i \cdot x) + b) \quad (7)$$

Several kernel functions have been used widely and successfully, such as, *polynomial kernel* with degree *d,*

$$K(x_i, x_j) = (1 + x_i \cdot x_j)^d \quad (8)$$

*Gaussian RBF kernel* with parameter $\sigma$,

$$K(x_i, x_j) = \exp\left(-\|x_i - x_j\|^2 / \left(2\sigma^2\right)\right) \quad (9)$$

and *sigmoid kernel* with parameter $\theta$,

$$K(x_i, x_j) = \tanh(x_i \cdot x_j - \theta) \quad (10)$$

### 3.2. Multi-Class Classification

SVMs are designed for binary classification. *K-class* classification problems can be solved using a *k-class* SVM which constructs a decision function by considering *k* classes altogether [13]. *K*-class classification problems can also are reduced to a collection of binary classification problems through several strategies, among which one-versus-rest strategy [13] and pairwise strategy [14] are often used.
The one-versus-rest method constructs *k* binary classifiers, one for each class. The *n*th classifier constructs a decision boundary between class *n* and the *k* - 1 rest classes. A testing example is classified in the class for which the distance from the margin in the positive direction is maximal [13].
The pairwise strategy creates $\left(\frac{k(k-1)}{2}\right)$ classifiers, one for each pair of classes. A majority voting is applied to make a decision for a testing example [14].
The comparison study in [13] shows the methods above to solve multi-class classification problems produce roughly similar accuracy.

## 4. EXPERIMENTAL RESULTS AND CONCLUSIONS

We use a texture database including 9 texture classes, 40 samples each. All images are in grayscale JPG format, each 640 by 480 pixels [15]. Each feature vector is composed of 41 image texture features among which 32 are derived from DCT coefficients and 9 from the ZMs. For each texture, 35 samples are used as training data in SVM and 5 samples are used as testing data.

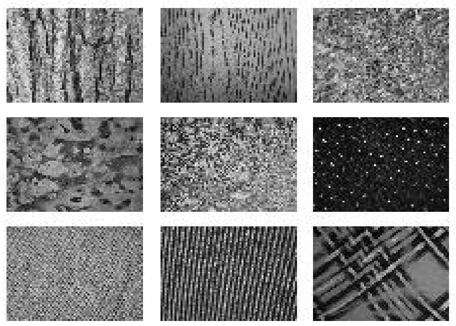

Fig.3. Experimental texture database.

We chose one-versus-rest method introduced in the section 3.2 to classify 9 textures, since it constructs much less classifiers than pairwise method (9 vs. 36 classifiers) but still can achieve good performance.

The image data are classified using SVM$^{light}$ developed by Joachims [16]. We chose RBF kernel with the parameter $\sigma$ of 0.001. The regularization parameter $C$ is set to 1. Table 1 shows the testing accuracies of 9 classifiers for the image data with the different levels of resolutions. The last row shows the testing accuracies by combining the decisions from 9 classifiers.

TABLE I
SVM TESTING ACCURACIES (%) OF 9 CLASSIFIERS USING ONE-VERSUS-REST METHOD FOR THE TEXTURES WITH DIFFERENT RESOLUTIONS

| **Classifier #** | **.125** | **.25** | **.5** | **.75** | **1** |
|---|---|---|---|---|---|
| 1 | 95.6 | 95.6 | 97.8 | 97.8 | 100 |
| 2 | 91.1 | 95.6 | 97.8 | 97.8 | 97.8 |
| 3 | 97.8 | 100 | 95.6 | 95.6 | 95.6 |
| 4 | 100 | 100 | 100 | 100 | 100 |
| 5 | 95.6 | 95.6 | 97.8 | 97.8 | 97.8 |
| 6 | 100 | 100 | 100 | 100 | 100 |
| 7 | 91.1 | 91.1 | 100 | 100 | 91.1 |
| 8 | 88.9 | 91.1 | 95.6 | 95.6 | 95.6 |
| 9 | 100 | 93.3 | 93.3 | 93.3 | 93.3 |
| Combined | **93.3** | **91.1** | **86.7** | **82.2** | **82.2** |

From Table 1 we can see that there is no apparent difference among the average binary classification accuracies when different image resolutions applied. The combined accuracies in the last row, which result from the one-versus-rest method, do show an interesting phenomenon: multi-class classification of texture images with low resolution achieves better classification accuracy.

In this paper, we define a multi-level DCT which applies DCT on the same signal with different levels of resolutions. MDCT coefficients combined with 9 Zernike Moments are used by SVMs for texture image classification. The experimental results suggest that texture image features extracting from its low resolution images by MDCT achieve both higher classification accuracy and less computing cost. This result coincides with the case we described in section 2.1 where a low resolution image can remove the redundant information resided in the image. This is especially true for the texture image with high resolution in which pixel intensity changes so slowly that the AC coefficients are all the zeros in most of blocks. In this case, image features will be mainly reflected by its DC component and the mean and standard deviation of the AC coefficients among blocks will be too 'flat' to distinguish themselves from the other texture classes.

## 5. REFERENCES

[1] Y. Rui, T. Huang, and S. Chang, "Image Retrieval: Current Techniques, Promising Directions and Open Issues," *J. of Visual Communication and Image Representation*, vol. 10, no.4, pp. 39-62, 1999.

[2] H. Li, G. Liu, and Z. Zhang, "A New Texture Generation Method Based on Pseudo-DCT Coefficients," *IEEE Transactions on Image Processing*, vol. 15, no. 5, 2006.

[3] B.S. Manjunath and W.Y. Ma, "Texture Features for Browsing and Retrieval of Image Data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no .8, pp. 837-42, 1996.

[4] S.E. Grigorescu, N. Petkov and P. Kruizinga, "Comparison of Texture Features Based on Gabor Filters," *IEEE Transactions on Image Processing*, vol. 11, no. 10, pp. 1160-1167, 2002.

[5] T. Weldon, W.E. Higgins and D.F. Dunn, "Efficient Gabor-Filter Design for Texture Segmentation," *Pattern Recognition*, vol. 29, no. 12, pp. 2005–2016, 1996.

[6] Y. Hamamoto and S. Uchimura, "A Gabor Filter-Based Method for Recognizing Handwritten Numbers," *Pattern Recognition*, vol. 31, no. 4, pp. 395–400, 1998.

[7] W.H. Chen, and W.K. Pratt, "Scene Adoptive Coder," *IEEE Transactions on Communications*, vol. 32, pp. 225-232, 1984.

[8] C. Min, S. Cho, K.W. Lim, and H. Lee, "A New Adaptive Quantization Method to Reduce Blocking Effect, *IEEE Transactions on Consumer Electronics*, vol. 44, no. 3, pp. 768-773, 1998.

[9] J.R. Smith and S.F. Chang, "Transform Feature for Texture Classification and Discrimination in Large Image Database," *Proc. IEEE Inter. Conf. on Image Processing*, vol. 3, pp.407-411, 1994.

[10] Y.L. Huang, "A Fast Method for Textural Analysis of DCT-Based Image," *Journal of Information Science and Engineering*, vol. 21, no. 1, pp. 181-194, 2005.

[11] X. Fu, Y. Li, R. Harrison, S.O. Belkasim, "Content-based Image Retrieval Using Gabor-Zernike Features," *The 18th International Conference on Pattern Recognition* pp.417-420, 2006

[12] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer-Verlag, New York, 1995.

[13] J. Weston and C. Watkins, "Multi-class support vector machines," *Technical Report CSD-TR-98-04*, Department of Computer Science, Royal Holloway, University of London, Egham, Surrey TW20 0EX, England, 1998.

[14] T. Hastie and R. Tibshirani, "Classifcation by Pairwise Coupling," *Technical Report*, Stanford University and University of Toronto, 1996.

[15] S. Lazebnik, C. Schmid, and J. Ponce, "A Sparse Texture Representation Using Local Affine Regions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1265-1278, August 2005.

[16] T. Joachims, Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning, B. Schölkopf and C. Burges and A. Smola (ed.), MIT-Press, 1999. Available: http://svmlight.joachims.org/