ACM 2nd International Conference on Multimedia, Oct. 1994, San Francisco, CA

QUAD-TREE SEGMENTATION FOR TEXTURE-BASED IMAGE QUERY

John R. Smith and Shih-Fu Chang

Center for Telecommunications Research and Electrical Engineering Department
Columbia University, New York, N. Y. 10027

ABSTRACT

In this paper we propose a technique for segmenting images by texture content with application to indexing images in a large image database. Using a quad-tree decomposition, texture features are extracted from spatial blocks at a hierarchy of scales in each image. The quad-tree is grown by iteratively testing conditions for splitting parent blocks based on texture content of children blocks. While this approach does not achieve smooth identification of texture region borders, homogeneous blocks of texture are extracted which can be used in a database index. Furthermore, this technique performs the segmentation directly using image spatial-frequency data. In the segmentation reported here, texture features are extracted from the wavelet representation of the image. This method however, can use other subband decompositions including Discrete Cosine Transform (DCT), which has been adopted by the JPEG standard for image coding. This makes our segmentation method extremely applicable to databases containing compressed image data. We show application of the texture segmentation towards providing a new method for searching for images in large image databases using "Query-by-texture."

1. INTRODUCTION

There is growing demand for database systems which support indexing, searching and retrieval of image, video and multimedia data. Approaches towards these databases have included:

- multiresolution image representations for browsing and retrieval at various scales [9][19],
- textual descriptions of data for keyword indexing [8],
- feature sets for searching by content, e.g., *texture*, color, shape, spatial relationships, etc. [2][16].

Multiresolution image representations allow the user to browse through databases by viewing thumbnail sketches of image data. They also enable incremental transmission and multiple decision points for data retrieval. But even with multiresolution browsing and retrieval, it is still difficult to search through large amounts of data. Keywords can be used to index the images in the database. However, this requires textual description of visual data, which in general can be neither complete nor consistent. The keyword approach cannot be automated because humans must view and interpret each image in order to ascribe text to it. Lastly, content-based approaches have been investigated which allow searching based on visual features of the image data. Here, the characteristics of the data which correspond to intuitive visual notions can be exploited using feature sets extracted from the data. This can allow searching to be performed using color-keys, texture-keys and shape-keys much the same way that keyword indexing is used.

This content-based approach towards image, video and multimedia databases allows the user to devise creative ways of formulating queries to produce both anticipated and serendipitous results. While this technique may not replace keyword and other indexing schemes entirely, it can certainly be used to compliment text-based schemes. We envision that content-based approaches to searching through image databases will add a new dimension of expressiveness for users to formulate queries and search for visual data.

2. CONTENT-BASED QUERY

We are currently developing the Content-Based Visual Query (CBVQ) System to provide access to image, video and multimedia databases using feature sets that describe the visual characteristics of the data. Furthermore, the CBVQ System will be part of the Video-On Demand (VOD) prototype being developed at Columbia University. In addition to supporting multiresolution retrieval and keyword indexing, the CBVQ and VOD systems will support searching based on image and video visual content, including texture, color and shape. Currently we are using image collections from the fields of medicine, art, photojournalism, astronomy, and movies on laser disc. The interface for the CBVQ System in browsing mode showing thumbnail sketches of art images appears in Figure 1.

FIGURE 1. Columbia University's Content-Based Visual Query System -- Graphical User Interface, showing collection of art images.

A major concern with implementing the texture query system is the availability of useful texture keys. Currently, our system provides users with three ways of utilizing keys for searching: by selection from collections of sample texture swatches, using texture synthesis and manipulation tools and using cutting tools for extracting regions of texture from images. The images may be obtained from results of previous retrievals or be provided by user.

There are many research issues involved with implementing this feature-based retrieval system, such as feature set identification, feature space compaction, image segmentation and multidimensional point indexing. This paper focuses on the image segmentation and indexing aspects. We make reference to other work of ours on texture feature set identification combined with feature space compaction [22].

Image segmentation is essential in the implementation of feature-based techniques for searching image databases. Effective segmentation will isolate the important homogeneous regions of the images in the database, from which an index can be established for searching. In most previous attempts at developing query methods for image databases, researchers have not chosen to automate the segmentation process [2]. Instead, they have relied on human intervention to mark the separate regions within each image. However, manual segmentation of all images in a database is a very time-consuming process. There will be much gained from a procedure that automates segmentation. However, automatic texture segmentation is a difficult problem. It requires unique mathematical operators that discriminate between an unlimited number of textures while providing precise identification of borders between textured regions. It has been found that these are conflicting objectives [22]. Therefore, in our approach to this problem, segmentation is only roughly estimated using a block-based approach based on quad-tree spatial decomposition of images. We maintain that spatial blocks of homogeneous texture will be sufficient in an image database system supporting "query-by-texture."

2.1 TEXTURE-BASED IMAGE QUERY

The visual characteristics of homogeneous regions of real-world images are often identified as texture. These regions may contain unique visual patterns or spatial arrangements of pixels which regional gray-level or color alone may not sufficiently describe. Typically, textures have been found to have statistical properties, structural properties, or both [13]. Moreover, due to the diversity of textures appearing in natural images it is difficult to give a universal definition of texture.

Texture is an important element to human vision. Belesz [15] has reported that a "preattentive visual system" exists which can almost instantaneously identify "textons", the preattentive elements of textures. The human visual system has also been found to decompose the retinal image into narrow bands of frequency and orientation which are important for discerning texture primitives [3]. Textures also have been used in 3-D visual recognition systems to provide cues to scene depth and surface orientation. In graphics systems, greater "realism" is achieved when textures are mapped to 3-D surfaces. Human beings also tend to relate texture elements of varying size to a plausible 3-D surface [1]. Texture features have been used to identify contents of ariel imagery such as bodies of water, crop fields and mountains. Finally, textures may be used to describe content of many real-world images: for example, clouds, trees, bricks, hair, fabric all have textural characteristics. Particularly when combined with color and shape information the details important for human vision are provided [6].

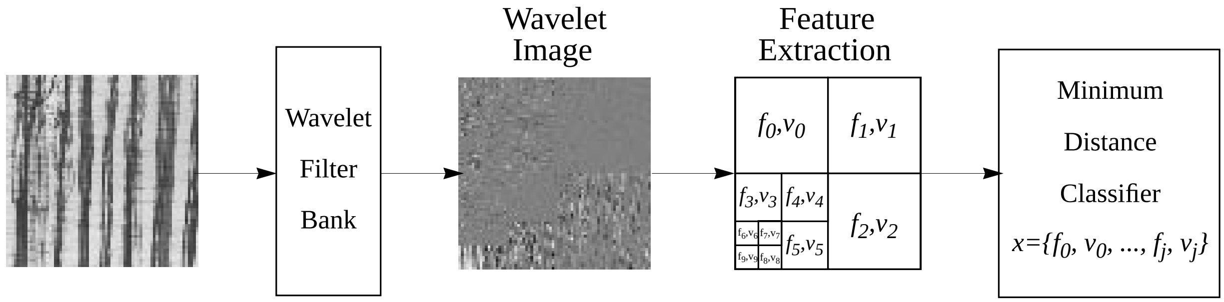

FIGURE 2. Texture classifier using wavelet subband features.

Because of the fundamental importance of texture information for human vision, texture can provide a meaningful tool for searching image databases. By indexing on the texture contents of images in the database, a user may search through large volumes of images using texture keys. Ideally, there is no restriction that the textures belong to predefined classes. Furthermore, the “Query-by-texture” can be combined with other descriptions of color and shape, to formulate overall image content-based queries.

2.2 TEXTURE DISCRIMINATION AND SEGMENTATION

Towards the development of “Query-by-texture,” it is necessary to find a meaningful measure of the similarity of textures (*texture discrimination*) and to develop a procedure for segmenting images based on textural content (*texture segmentation*). Furthermore, discriminant functions are needed to gauge the similarity between textured regions and the texture key used for searching. Likewise, the discriminant functions can be used to match blocks within each image to produce the *texture segmentation*.

Using a quad-tree data structure, we carry out *texture segmentation* by grouping image spatial blocks. In our approach, all feature sets are computed directly from the Quadrature Mirror Filter (QMF) wavelet representation of the images. While the QMF wavelet features have been found to provide good classification of Brodatz textures, we found in our previous work [20], that feature sets derived from other image representations, such as DCT and uniform subband representation, are also effective. Since an image database may contain hundreds of thousands of images, it is beneficial when segmentation can be performed using data directly available in the given image representations. We have focussed on QMF wavelet features keeping in mind the application to large image databases. Although, our approach is general enough to use instead other features sets that discriminate between textures.

Because the goal of this segmentation is to provide indexing of images in the database, we relaxed the constraint that the segmentation provide perfect boundary identification. Our segmentation is successful when representations of the important regions of texture of the images are obtained. Using the spatial quad-tree approach, each quad-tree node points to a block of image data. Children nodes are merged when the discriminant functions indicate that the children blocks contain sufficiently similar textures. A “Query-by-texture” examines the blocks identified by the final quad-tree structures to test the similarity to a texture-key used for the search.

3. TEXTURE FEATURE EXTRACTION

3.1 BLOCK-BASED FEATURES

Some approaches to texture segmentation fall under the category of pixel-based schemes. In general, pixel-based segmentation schemes evaluate the texture features in a neighborhood surrounding each pixel in an image. When neighboring pixels are classified similarly, regions of texture are formed. Of course, a difficulty results when image pixels border several textured regions. Then the features may not resemble those of the nearby textures. Sophisticated techniques are often adopted to decipher this border information between textures. However, in an image database application, precision in border extraction may not be necessary as long as a block of each texture is identified. Since the blocks of texture may suffice for matching, we adopt a computationally less expensive quad-tree approach for image segmentation which can use traditional texture discriminant functions to group blocks within the image. By utilizing a block-based approach towards feature extraction, the loss in boundary localization is a direct result of the uncertainty principle [25]. However, since we use blocks of data from which to compute texture features, the spatial localization can be traded off for better spectral selectivity and statistically computed features.

3.2 WAVELET SUBBAND FEATURES

Effectively, we treat each block as a separate image of texture. Any features extracted from the blocks that provide discrimination may be used. The features we use are computed from the mean absolute value and variance measures

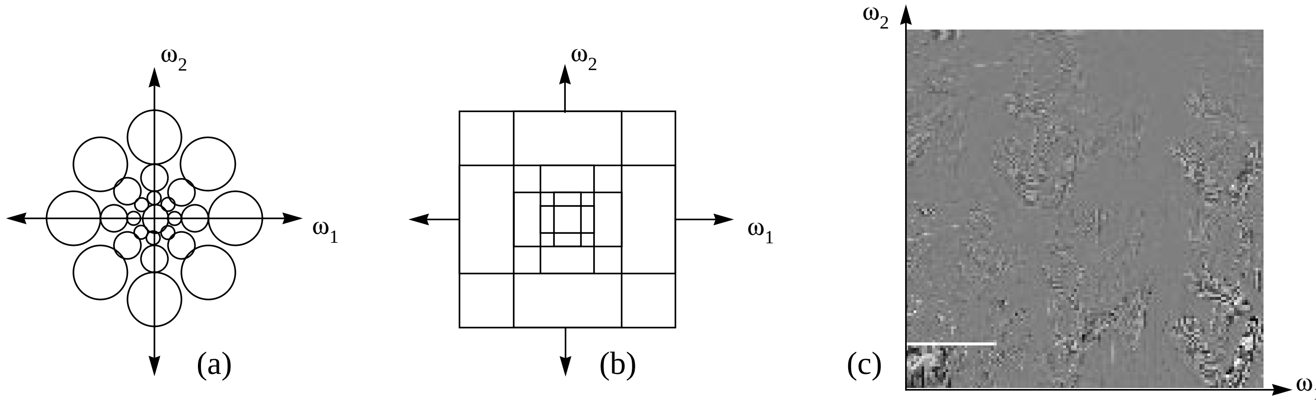

FIGURE 3. (a) typical Gabor filter tilings used for texture discrimination, and (b) frequency tiling produced by orthogonal separable (QMF) wavelet decomposition, and (c) QMF wavelet decomposition of the Barbara image.

on the subbands produced from three iterations of QMF wavelet decomposition. This feature set was found to give 93% correct classification of the complete set of 112 Brodatz textures [22]. The wavelet feature-based texture classification process is illustrated in Figure 2.

The QMF wavelet subband decomposition is an orthogonal approximation to that produced by Gabor filters. Gabor filters have been applied to texture segmentation by Bovick [4] and Jain [14]. The advantage of these wavelet spatial-frequency approaches is that simple statistics computed from the subband images may be used because the subbands have limited spectral information [14]. Gabor filters have found particular application in texture discrimination because a filter set can consist of arbitrarily scaled and rotated filters. Gabor filter banks have also been found to approximate the mechanisms of human vision [14]. The filters also meet the minimum combined uncertainty in the spatial and frequency domains. Most Gabor filter banks used for texture discrimination do not provide a complete basis for decomposition as shown in Figure 3(a). The QMF wavelet decomposition does provide an orthogonal basis but does not provide as much granularity for identifying frequency directional components as shown in Figure 3(b). A QMF wavelet decomposition of the Barbara image appears in Figure 3(c).

We combine the quad-tree data structure with wavelet subband representation to perform image segmentation as follows: for each quad-tree block, features are extracted by computing the wavelet decomposition of the block, as indicated in Figure 4(a). Features are obtained by measuring statistics from the wavelet subbands. However, in general, to perform wavelet decomposition, image borders require padding before filtering is carried out. Typically, padding information is taken from within the image based on some rule, such as mirror extension of borders. Since we are dealing with image blocks and not actually separate image entities, the border information can be extracted from neighboring image blocks by borrowing border pixels in the filtering operation.

Done this way, the wavelet decomposition and feature extraction processes are no longer independent for each spatial block. But the exchange offers an elegant solution for padding, and the order of operations can be reversed. First, the entire image is decomposed using wavelet filtering, then quad-tree spatial blocks point to appropriate regions in the full wavelet image, as shown in Figure 4(b). Furthermore, features can be extracted directly from the wavelet images, if the wavelet representation is used to store images in the database. We placed some focus on this representation since the wavelet representation has a number of beneficial qualities in the image database application, such as providing a multiresolution structure and energy compaction to enable compression.

4. TEXTURAL SIMILARITY

4.1 TRAINING

The Brodatz texture collection was used to obtain discriminant functions using Fisher Discriminant Analysis. This procedure constructs linear composites of the features which provide for maximum average separation among training classes [10]. In our previous work, discriminant functions generated from a fractional subset of Brodatz textures were found to provide effective discrimination among the whole set [22]. We hope that using the complete set of 112 Brodatz classes in training will construct discriminant functions general enough to discriminate between new and unknown textures. The Mahalanobis [12] distance, which is the squared distance in the transformed feature space, was used to measure the similarity between textures. In ordinary

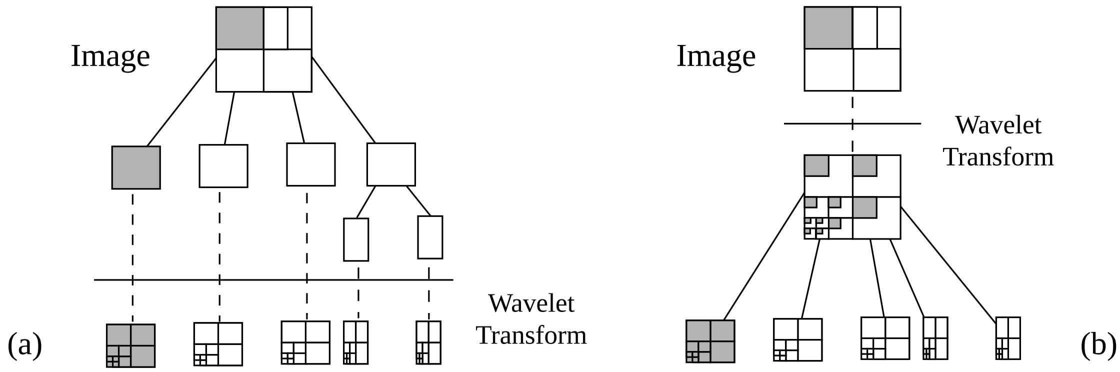

FIGURE 4. (a) Quad-tree decomposition followed by wavelet-based feature extraction performed on quad-tree blocks, (b) Quad-tree blocks point to regions in wavelet image for feature extraction.

classification of textures or comparisons of many textures, the relative rankings of the Mahalanobis distances are used to identify closest matches. Subsequently, in response to a "Query-by-texture," all textures in the database will be sorted by this distance to the texture-key. This search requires no threshold to be established to determine when textures are no longer "similar." However, to decide whether two textures are similar or not, as in quad-tree segmentation, requires a threshold in distance. In general it can be difficult to set this threshold for all textures. We have used observations made on the distributions of the 112 Brodatz texture classes in feature-space to estimate an effective threshold for similarity between textures.

4.2 DISTANCE THRESHOLD

Using a fixed distance threshold for determining whether two textures are sufficiently similar will not be optimal for all types of textures. Depending on the characteristics captured by the extracted feature sets and the particular derived non-singular mapping to a transformed feature space, the within-class variance for visibly similar textures will vary from class to class. We have found that the distance threshold depends heavily on the block size from which the features are extracted and on the energy of the feature set.

4.2.1 BLOCK SIZE

There is a relationship between block size and the estimation of the features probability distribution produced from the blocks. Since smaller texture blocks will contain fewer data points from which to derive statistical features, there will be a larger deviation in features extracted from these blocks of similar textures. This results in a greater variance in distance between possibly similar textures, necessitating a higher distance threshold for comparing textures of smaller block size. Likewise, features extracted from larger blocks produce smaller within-class variation, necessitating a lower distance threshold for comparing textures of larger block size.

4.2.2 FEATURE SET ENERGY

The within-class variation is also correlated to the magnitude of the transformed feature sets, as depicted in Figure 5(a). For a texture class i, the distance d_i in feature space for a member of that class to the centroid χ_i of that class is a function of the energy of the feature set $\|\upsilon_i\|$. Textures that produce high energy features sets typically belong to classes with the largest within-class variance. Likewise, textures producing low energy features typically belong to texture classes with small within-class variance. Figure 5(b) shows the energy plot of four Brodatz textures that illustrate this relationship. The distance threshold can be determined from information about the textures being compared, namely image region size and energy of the feature set.

4.2.3 DISTANCE THRESHOLD COMPUTATION

To develop the distance threshold to be used for quad-tree segmentation, we examined the training texture cuts taken from Brodatz collection previously, and observed a strong positive correlation (0.75) between the quotient of feature energy and image size, and distance to correct class. Performing a linear regression analysis on this data, the threshold function in EQ 1 was obtained. Here τ is the threshold to be used in the Mahalanobis distance, $\|\upsilon\|$ is the energy of the transformed feature vector and s is the number of pixels in the image block.

$$\tau = 82.057 \cdot \frac{\|\upsilon\|}{s} - 1.748 \qquad \text{(EQ 1)}$$

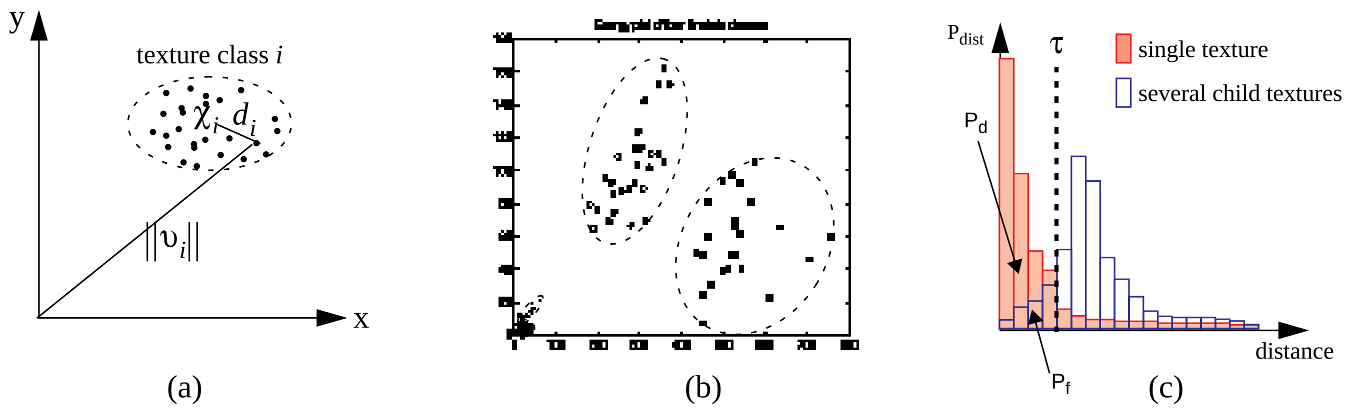

FIGURE 5. (a) the distance d_i of each texture class member to the centroid χ_i of the class is a function of energy of the feature set $\|\upsilon_i\|$, (b) first two most significant dimensions in feature space for four random Brodatz texture classes, (c) histograms of the sums of distances of children to parent when one single homogeneous texture present and more than one texture present

4.2.4 THRESHOLD FOR QUAD-TREE SPLITTING

Beginning with the complete image as the first node, the quad-tree is formed by iteratively splitting each node into children nodes. Four children are spawned by each parent and based on the texture features extracted for each child, conditions for merging are tested. The distance threshold is computed for each child as a function of the child feature energy and block size using EQ 1. Then the distances in feature space are measured from the parent node to each child. If the distances to all four children fall within the respective thresholds of the children, a single texture is declared in the parent node, as illustrated in Figure 6(a). Then the parent is marked as a terminal node and the children are deleted. If a single texture is not present, then tests for pair-wise grouping of children are performed. When the distance between the next two closest children falls below both respective thresholds, the children are merged, see Figure 6(b). When no children are close, all children are kept as nodes, and quad-tree iteration continues on each child, see Figure 6(c).

4.2.5 LIKELIHOOD RATIO TEST

In actuality the value of the distance threshold determines the trade-off between P_d, the probability of correct detection of similar textures and P_f, the probability of false detection. Due to the nature of the underlying distributions of parent-child distances when a single textured region is present versus the case when more than one texture among children blocks is present, P_d and P_f will always increase and decrease together. This can be observed in the histogram plots of parent-child distances for the two cases presented in Figure 5(c). Furthermore, a result of signal detection theory tells us that using a likelihood ratio test, P_d can always be higher than or equal to P_f [24]. Therefore, an alternative to formulating the distance threshold formula from training data as in the previous section, is to estimate the a priori probability distributions of distances between parent-child blocks. Then the threshold can be shifted to adjust the trade-off between P_d and P_f. This framework allows the capability to adjust the texture segmentation to the optimal trade-off between P_d and P_f, given some cost functions for each.

5. INDEXING IMAGES BY TEXTURE

5.1 QUERY-BY-TEXTURE

An image database was generated by randomly compositing five cuts per image from 134 images consisting of 112 Brodatz texture images and 22 real world images to generate a database of 200 images. To generate the composite images, first four random pieces were tiled in each of the corners to cover the entire image area. Then a fifth random cut was positioned in the center, overlapping parts of the four cuts. Finally, a random shift was applied in both the x and y directions with a wrap around of image data to scramble the blocks of texture. This produced images each with five possibly distinct textures contained in possibly many disjoint regions. “Queries-by-texture” were then performed on this composite image database using cuts from the Brodatz textures as texture keys. A typical result from a “Query-by-texture” on this composite texture image database is shown in Figure 7.

6. SEARCHING FEATURE SPACE

6.1 HIERARCHICAL SEARCHING

As the number of images and textured regions increases, it becomes necessary to utilize a searching procedure through

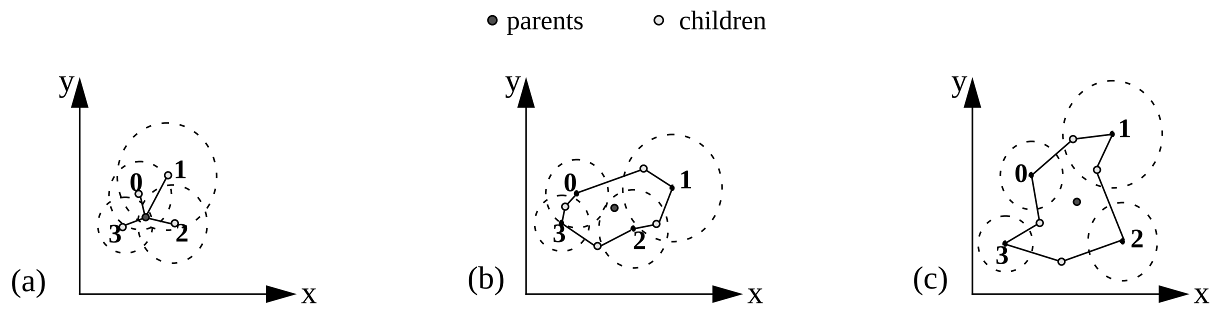

FIGURE 6. Splitting examples. Note: circle radius = distance threshold for each child as computed as a function of child energy and block size. (a) No Split -- all children belong to single class, (b) Split into two -- children 0 and 3 form a texture class, and children 1 and 2 form a second texture class, (c) Split into four -- four texture classes present in this spatial block.

the database more efficient than exhaustive search. The ordering of the discriminant functions by significance is a natural by-product of the feature-space compaction performed by the Fisher Discriminant Analysis. This energy compaction may be exploited by adapting a hierarchical approach to searching. Therefore, each dimension of the feature space may by searched independently in order of significance. For example, a binary search may be conducted on sorted feature data beginning with the most significant discriminant functions. At first, the candidate match list may be kept larger than the final desired number of retrievals. As the list is subsequently searched using the remaining discriminant functions, it may be further truncated. This technique of successive refinement reduces the complexity of searching the high-dimensional feature space by reducing the overall number of comparisons.

6.2 FEATURE SPACE PARTITIONING

A second alternative to searching the high-dimensional texture feature space is to partition the feature space. This way the search can be conducted in two stages. The first will identify the appropriate partition in feature space. The next stage will examine only the data points in the partition. However, if the dimension of the feature space is large, it is unlikely that uniform partitioning will be optimal. For example, data may crowd into several partitions while others remain sparse or empty. This will result in non-uniform search performance.

6.3 DATA CLUSTERING

An improvement over partitioning the feature space is to gather data points into clusters. Again the search will be broken into two stages. The first will identify the appropriate cluster, and the second will search all the data points in that cluster. Search performances at each stage can be regulated by the clustering algorithm. For example, a bound set on the maximum number of data points per cluster will produce a bound on the second stage of the search algorithm. This results in more consistent performance than partitioning the feature space.

7. CONCLUSION

We have presented a method for performing “Query-by-texture” on an image database. Texture has been shown to be a viable characteristic of visual information by which image data may be indexed in a large image database. Using a quad-tree approach to image segmentation, feature sets are extracted from image blocks and conditions for merging are determined by a texture discriminant function. Without resolving border details between textured regions, we use the homogeneous rectangular blocks of texture within each image to perform indexing in the database. Our approach, in general, places no limitations on how features are extracted from image blocks, nor does it require that the features even be of texture. We have used specifically features sets based on the QMF wavelet decomposition because of the discrimination performance, and the benefits this representation offers in a database application. The quad-tree method offers an efficient approach towards segmenting and representing textures present in an image, and provides a general framework by which other discriminating features can be used for image segmentation.

7.1 FUTURE WORK

We are currently extending the texture segmentation and “Query-by-texture” methods to a database of real-world images that consist of images from nature, art, architecture and medicine. The “Query-by-texture” feature will be integrated with keyword indexing to provide the user with both textual and visual keys for searching for images in the database. We will also be exploring other features such as color and shape to be used to perform overall content-based queries of image databases. This work will be part of the Content-Based Visual Query System and Video On-Demand prototype efforts at Columbia University [7].

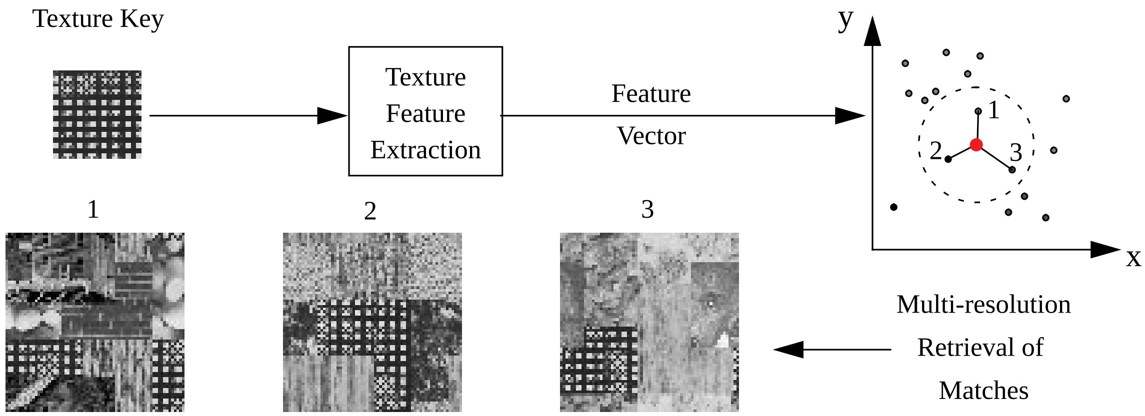

FIGURE 7. "Query-by-Texture" -- using a cut from Brodatz texture D101 as a texture-key to search through the image database. The three closest matches shown at the bottom are returned to user using multi-resolution retrieval.

REFERENCES:

[1] Dana H. Ballard and Christopher M. Brown, *Computer Vision*, Prentice-Hall, Inc., 1982.

[2] R. Barber, W. Equitz, M. Flickner, W. Niblack, D. Petrovic, P. Yanker, "Efficient Query by Image Content for Very Large Image Databases," *COMPCON*, San Francisco, 1993.

[3] Jacob Beck and Anne Sutter and Richard Ivry, "Spatial Frequency Channels and Perceptual Grouping in Texture Segregation," *Computer Vision, Graphics, and Image Processing*, 1987, Vol. 37, pp. 299--325.

[4] A. C. Bovick, "Analysis of Multichannel Narrow-Band Filters for Image Texture Segmentation," *I.E.E.E. Transactions on Signal Processing*, Vol. 39, No. 9, September 1991, pp. 2025 - 2043.

[5] P. Brodatz, Textures: a Photographic Album for Artists and Designers, Dover, New York, 1965.

[6] Terry Caelli and David Reye, "On the Classification of Image Regions by Colour, Texture and Shape," Pattern Recognition, Vol. 26, No. 4, pp. 461--470, 1993.

[7] Shih-Fu Chang and Dimitri Anastassiou, "Development of an Advanced Image/Video Server in the Video on Demand Testbed," IEEE Workshop on Visual Signal Processing & Communications, New Jersey, September, 1994.

[8] S. K. Chang, C. W. Yan, D. C. Dimitroff and T. Arndt, "An Intelligent Image Database System," *I.E.E.E. Transactions on Software Engineering*, vol. 14, no. 5, May 1988.

[9] N. Dal Degan, R. Lancini, P. Migliorati and S. Pozzi, "Still images retrieval from a remote database: The system *Imagine*," *Signal Processing: Image Communications*, vol. 5, 1993, pp 219 - 234.

[10] William R. Dillon and Matthew Goldstein, *Multivariate Analysis*, John Wiley & Sons, 1984

[11] C. Faloutsos, et. al., "Efficient and Effective Querying by Image Content," IBM RJ 9453 (83074), August 3, 1993.

[12] Keinosuke Fukunaga, *Introduction to Statistical Pattern Recognition*, Harcourt Brace Javanovich, 1990.

[13] Robert Haralick, "Statistical and Structural Approaches to Texture," *Proceedings of the I.E.E.E.*, May 1979, Vol. 61, No. 5, pp. 786--803.

[14] Anil K. Jain and Farshid Farrokhia, "Unsupervised Texture Segmentation Using Gabor Filters," *Pattern Recognition*, Vol. 24, No. 12, pp. 1167--1186, 1991.

[15] B. Julesz and J.R. Bergen, "Textons, The Fundamental Elements in Preattentive Vision and Perception of Textures," 1983, pp. 243--456.

[16] Wayne Niblack, "Querying large image databases by content," *OE Reports of The International Society for Optical Engineering*, No. 123, March 1994, pp. 15.

[17] Wayne Niblack, et. al., "The QBIC Project: Querying Images by Content using Color, Texture, and Shape," IBM RJ 9203 (81511), February 1, 1993.

[18] Philippe P. Ohanian and Richard C. Dubes, "Performance Evaluation for Four Classes of Textural Features," *Pattern Recognition*, Vol. 25, No. 8, pp. 819--833, 1992.

[19] A. Ortega, Z. Zhang and M. Vetterli, "Modeling and Optimization of a Multiresolution Remote Image Retrieval System," INFOCOM 1994.

[20] Todd R. Reed and J.M. Hans du Buf, "A Review of Recent Texture Segmentation and Feature Extraction Techniques," CVGIP:Image Understanding, Vol. 57, No. 3, May 1993, pp. 359--372

[21] Hanan Samet, *Applications of Spatial Data Structures -- Computer Graphics, Image Processing, and GIS*, Addison - Wesley Publishing Company, 1990.

[22] John R. Smith and Shih-Fu Chang, "Transform Features for Texture Classification and Discrimination in Large Image Databases," accepted for I.E.E.E. International Conference on Image Processing (ICIP) in Austin, Texas 1994.

[23] M. Spann and R. Wilson, "A Quad-Tree Approach to Image Segmentation which Combines Statistical and Spatial Information," *Pattern Recognition*, Vol. 18, No. 3/4, pp. 257--269, 1985.

[24] Harry L. Van Trees, Detection, Estimation, and Modulation Theory, John Wiley & Sons, New York, 1968.

[25] Roland Wilson and Michael Spann, *Image Segmentation and Uncertainty*, Research Studies Press, John Wiley & Sons, New York, 1988.