Extracting Multi-Dimensional Signal Features for Content-Based Visual Query

Shih-Fu Chang and John R. Smith

Department of Electrical Engineering & Center for Telecommunications Research
Columbia University, New York, NY 10027, U.S.A.

ABSTRACT

Future large visual information systems (such as image databases and video servers) require effective and efficient methods for indexing, accessing, and manipulating images based on visual content. This paper focuses on automatic extraction of low-level visual features such as texture, color, and shape. Continuing our prior work in compressed video manipulation, we also propose to explore the possibility of deriving visual features directly from the compressed domain, such as the DCT and wavelet transform domain. By stressing at the low-level features, we hope to achieve generic techniques applicable to general applications. By exploring the compressed-domain content extractability, we hope to reduce the computational complexity. We also propose a quad-tree based data structure to bind various signal features. *Integrated feature maps* are proposed to improve the overall effectiveness of the feature-based image query system. Current technical progress and system prototypes are also described. Part of the prototype work has been integrated into the Multimedia/VOD testbed in the Advanced Image Lab of Columbia University.

Keywords: Content-based image query, feature extraction, compressed-domain image manipulation and indexing, texture, color, shape feature extraction, image database and storage.

1. Introduction

As massive amounts of images and video are being produced everyday at a rapid speed, the challenging task of designing advanced Visual Information Systems (VIS) to manage and process these visual data is required. Effective extraction of visual features or contents is needed to provide meaningful index of and access to visual data. Most existing approaches to image indexing and retrieval use the *textual keyword* [4,5]. Search and retrieval are performed on the keyword records and the associated images are retrieved after the textual search is complete. Some image databases provide enhancement by supporting query by pictorial examples for constrained applications, such as geographic satellite images, mechanic design diagrams, and human facial images [6,11,28]. Enhancement can be achieved by using the so-called "iconic indexing" concept to explore the spatial relationships among image objects [12]. Also mentioned in the literature is the semantic level descriptions, such as "person A in front of a stone house" or "a high-speed racing car". Contents at this level usually require user input but usually are not complete nor consistent (e.g., different users may have different interpretations). Also, the vocabulary used in describing image contents is usually domain-dependent.

This study takes a different approach based on image processing and computer vision technologies. We will explore generic low-level visual features for applications of content-based visual query. Instead of asking users to specify image content by text, we would like to support image access systems which allow users to search through visual databases by using fundamental feature sets derived from object *texture, color, shape, and motion*, etc. These visual features are at a relatively low level, compared to the semantic content mentioned above. Although closer to human subjective perceptions, high-level visual semantic content is still very difficult for computers to extract without user assistance. By starting from the low-level visual features, we hope these features can be more generic and less constrained for potential applications. More importantly, automatic extraction algorithms of these visual features will remain feasible. Automatic mechanisms without substantial user involvement are essential for many image database and storage applications, such as satellite picture databases and large multimedia digital libraries. The concept of Query-by-Image-Content (QBIC) has been successfully and comprehensively demonstrated in [24]. However, user

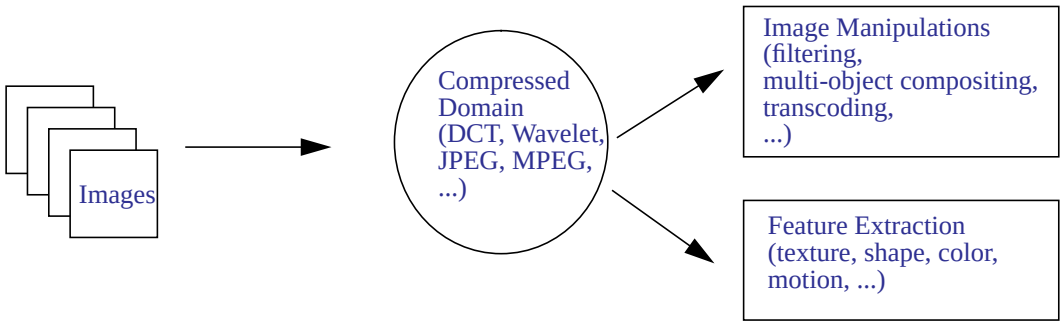


FIGURE 1. Concept of Compressed-Domain Image Manipulation and Feature Extraction.

manual input was used in extracting image content. Also, synergistic relations between feature extraction and image compression were not explored either.

Another critical technological requirement for VIS is image compression. Traditionally, image compression has focused solely on the compression performance factors, such as compression rate and computational complexity. In the context of VIS design, we believe considerations of image compression and feature extraction should be integrated together. This actually fits closely the recent trend of image coding effort in the research community [29]. In this paper, we will describe techniques for extracting visual signal features directly in the compressed domain and binding various features in a unified representation platform. This compressed-domain concept has been applied in image manipulations and filtering for different video applications [1]. Figure 1 illustrates the concept of compressed-domain image technology.

Our focus on low-level signal features does not exclude the usefulness of existing content description and indexing mechanisms such as keywords and user-provided semantics. Instead, we believe that a great multiplicity of query/indexing mechanisms including high-level semantics, and low-level visual features should be provided to accommodate the challenging needs of image indexing and access.

We first discuss various feature extraction techniques for texture, color, and shape. We describe a quadtree representation for binding these visual features. An integrated feature map is then proposed to integrate different visual features to improve the usefulness of feature-based image query. For texture and shape, we also explore the capability of extracting features from the compressed-domain, such as the DCT and wavelet transform domain. Finally, current prototype status is described.

2. Extracting Low-Level Visual Features — Texture, Color, and Shape

Feature extraction and pattern recognition has been extensively studied for decades. However, application to content-based image indexing and query is relatively new. In the context of VIS applications, new constraints and requirements have been imposed. First, image query and retrieval is basically based on *similarity matching*, instead of exact matching or classification. Exact matching has been typically used in information retrieval for traditional text information systems. Classification has been a typical task in computer vision and robotics where prior knowledge of finite expected objects is available. In the VIS context, image matching is based on ranking of similarity between all candidate images and the input image sample. The number of image classes is generally unlimited, except for specific constrained applications.

Secondly, as described earlier, feature extraction for image database applications needs to take into account other required image technologies, such as *image compression*. Many image compression techniques actually have already performed some functions of information filtering and signal decomposition that may be very suitable for feature definition. Therefore, it provides tremendous advantages if visual features can be extracted from the compressed domain directly. Note that besides the above mentioned advantage, the compressed-domain approach also greatly

reduces the required computational cost, since compressed images do not need to be decoded back to the uncompressed form.

Third, although accurate object segmentation will be very useful for image indexing if it is robust and efficient, current image processing technologies still fall far behind this goal. However, for the purpose of image indexing and retrieval, we think that fully accurate object segmentation may not be essential. We think it will suffice to some extent if "*prominent regions*" with "*distinctive features*" can be extracted and indexed. For example, regions with clear, distinctive textures may well point to textiles, terrains, and biological tissues. The important point is that these regions do not have to correspond to real objects. Based on this region-based low-level feature extraction approach, we hope to achieve the optimal comprise between automatic mechanisms and image content extraction.

Finally, future large VIS requires *efficient data structure* and feature comparison techniques in order to provide reasonable responsiveness to image query. Exhaustive search of every candidate image in a huge image database simply cannot be acceptable. Also, the first constraint mentioned above makes it difficult to find a data structure for indexing because of the similarity matching and the high dimensionality.

With the above considerations, we describe in this section techniques for extracting visual features like texture, shape, and color. We also describe the technique using a modified quad-tree data structure to bind these different visual features.

2.1 Texture

Texture is an important element to human vision. 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 of image objects important for vision are provided [19]. By identifying textural contents of images in the database, a user may search through large volumes of images using a texture key. A "Query-by-texture" can be formulated to search through the image database, returning images found to contain regions of similar texture. To accomplish this task, we need to solve two problems, first: how to differentiate different textures (texture discrimination) and how to localize image regions with distinctive textures (texture region extraction).

Texture Discrimination:

Techniques for describing textures can be roughly divided into two categories — statistical and structural. Another dimension for classifying texture discrimination techniques is feature-based vs. model-based [31]. Ad hoc methods also exist for describing textures, such as fractals. Psychophysical studies have shown that humans perceive textures by decomposing signals into components with different frequency and orientation. Gabor filter banks have been used to approximate the mechanisms of human vision in texture discrimination [20,21]. However, Gabor filter banks are neither complete nor orthogonal. In order to explore the maximum synergistic relationship between texture indexing and image compression, we use the feature sets defined in transform decomposition to approximate the feature extracted from Gabor filter banks. Transform decomposition of images can be obtained by taking discrete cosine transform (DCT), subband transform, or wavelet transform. From the decomposed signal bands, texture feature sets are extracted by measuring the subband energy. For example, for a 5-level wavelet decomposition, feature vectors with 16 terms are produced. Figure 1 illustrates the texture feature extraction procedure.

In order to get texture classes as complete as possible in our experimentation, we obtain the complete set of 112 Brodatz texture images [9]. We hope that using the complete set of Brodatz textures in training will construct a discriminant function general enough to discriminate between new and unknown textures. From the feature sets defined above, we use the Fisher Discriminant Analysis technique to achieve the maximum average separation among different texture classes [22]. The Mahalanobis distance (EQ 1) in the transformed feature space (i.e. after Fisher Discriminant Analysis) was used to measure the similarity between textures. In ordinary classification of textures or comparisons of many textures, the relative ranking of the Mahalanobis distances is used to identify closest matches. In response to a "Query-by-texture," all textures in the database will be sorted by distance from the texture-key.

Using the Fisher Discriminant Analysis technique, we were able to capture the feature elements with the maximum capability of texture separation. The criterion is to maximize class separability by calculating several scatter

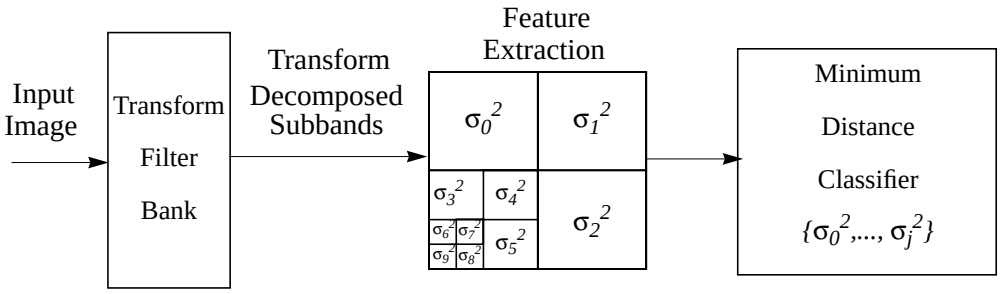


FIGURE 2. Texture feature extraction in the wavelet domain. $\{\sigma_j^2\}$ = {subband energy}

matrices, W (within-class), B (between class), and their "ratio" $W^{-1}B$. Feature elements are mapped to the eigenvectors with the maximum separability significance (i.e. eigenvalues of $W^{-1}B$). The classification rule is as follows — allocate x to class k if, for all $i \neq k$,

$$\sum_{j=1}^{s} (e'_j (x - \bar{x}_k))^2 \leq \sum_{j=1}^{s} (e'_j (x - \bar{x}_i))^2 \qquad \text{(EQ 1)}$$

where e_j is the selected eigenvectors, x_i is the representative feature vector for class i.

Figure 3 shows the correct classification rate versus the number of feature elements used. Even with only 6-8 feature elements, the classification rate still can maintain at the 90% level. The experiments were done for a Brodatz texture database populated from the original Brodatz test set. From each Brodatz class, about 20 image cuts with random size and position were generated. This resulted in a texture database with more than 2000 texture images from the complete set of Brodatz texture collection.

One of our research objectives was to evaluate the suitability of different compression algorithms. Figure 3 also shows a comparison of different compression algorithms based on the effectiveness of texture discrimination. It can be seen that the wavelet transform has the highest classification rate compared to the uniform subband and the DCT/Mandala transform[1]. The energy leakage problem associated with the DCT transform may be used to

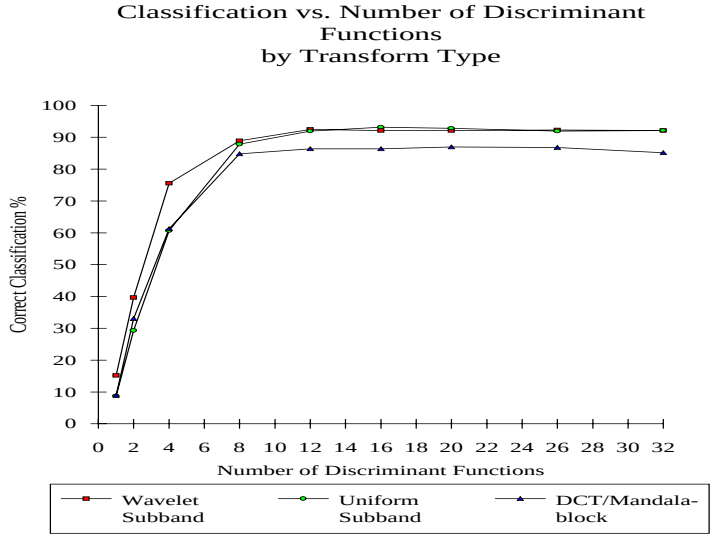


FIGURE 3. Effect of the number of feature elements used for classification on classification rate for the Brodatz texture database.

1. DCT/Mandala subbands can be obtained by re-ordering the DCT coefficients based on the coefficient ordinates [32]. It can be interpreted as DCT followed by a polyphase transform as well.

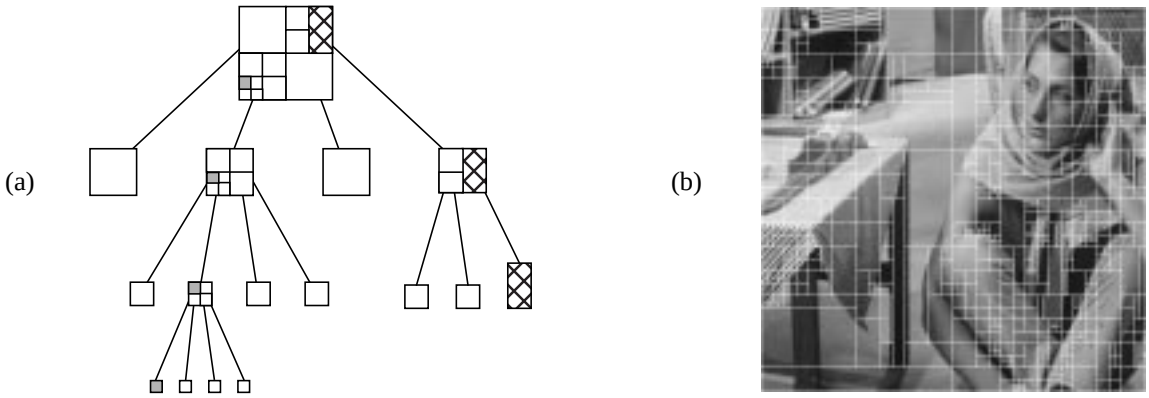


FIGURE 4. (a) Modified Spatial Quad-tree representation -- each tree parent can have two, three or four children, (b) Texture-Based Quad-tree segmentation of the Barbara image. Each Quad-tree node is indicated by white bordered region.

explain its inferiority in texture discrimination. However, the fact that DCT has been used popularly and commercially in many international image coding standards may still make DCT an attractive solution.

Modified Quad-Tree Based Texture Region Segmentation:

To extend the texture-based query approach to arbitrary local object search (as opposed to full image matching only), images in the database are segmented into textural regions, each of which has homogeneous texture features. The discriminant function mentioned above is used to match neighboring blocks within each image to perform the *texture segmentation*. Because the goal of this segmentation is to provide indexing of images, we relax the constraint that the segmentation provide perfect boundary extraction. Our texture segmentation is successful when texture regions of the images are represented accurately enough to provide the expected matches upon a "Query-by-texture". More accurate boundary information can be obtained in a boundary-sensitive feature such as shape, as described later.

Using the spatial quad-tree approach, each quad-tree node points to a block of image data. Children nodes are merged when the discriminant function indicates that the children blocks contain sufficiently similar textures. To decide whether two textures are similar or not, we have established some optimal thresholds. We have also modified the quad-tree structure to allow each parent node to have two, three or four children. When all four children cannot be merged together, subsets of the children may be paired horizontally or vertically depending on which arrangements group the most similar children. The quad-tree based region extraction process is also illustrated in Figure 4.

In the task of texture classification and discrimination, the above discriminant function is used to find the texture class with the highest similarity with the input texture key. No thresholds are required for decision making. However, for texture region extraction, the optimal texture distance threshold is required in order to merge neighboring image blocks which have sufficient similarity. Choosing optimal distance threshold is not trivial. A fixed distance threshold will not be optimal for all types of textures. From the experiment results with the Brodatz texture database, we found that the optimal distance threshold is correlated to the energy of the transform feature vector, but negatively correlated to the image block size [2].

The envisioned texture-based image indexing technique is to use the extracted texture regions and their attributes as image content indices. Figure 5 shows examples of the texture regions we have extracted from two natural images. Once we have these regions extracted, we can also index their derived attributes such as coordinates, size, orientation, and spatial relations among different regions.

In real applications of texture-based query, texture sample keys are supposed to be provided as examples from users. Simple editing/cutting tools may be used to select arbitrary regions from existing displayed images. Another approach is to provide texture synthesis tools which will allow users to adjust texture description/synthesis parameters. Currently, we are testing the usefulness of this texture-based indexing/query technique in practical applications such as satellite picture databases, medical image databases, and art image archives.

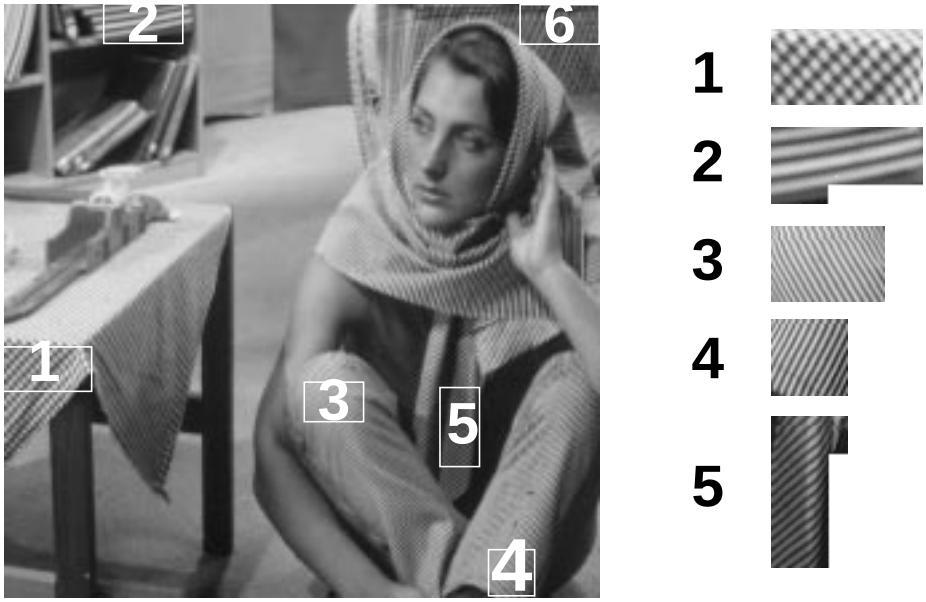


FIGURE 5. The results of modified quad-tree based texture region extraction. Significant texture regions with homogeneous texture content are abstracted out as the indices of images in the database.

2.2 Color

Psychophysical research has been studying how human vision system discriminates different colors. There has been research applying color features as query keys to image database applications. Binaghi *et al.* have developed a system which uses only color as the index into a database of color fabric samples [8]. Niblack *et al.* includes color feature in their QBIC system [24]. Swain *et al.* used color indices for 3D object classification [23].

Color-based indexing provides several unique advantages. It's less sensitive to noise and background complication, compared to other features such as shape and textures. Also, it is independent of image size and orientation. However, it also has disadvantages such as high sensitivity to illumination and shade. Furthermore, the choice of color representations (*i.e.*, color space) is very critical. Different color spaces will result in different color dynamic ranges, different color differentiation capability. The popular RGB color space is efficient for display, but inappropriate for color feature indexing and discrimination. One important criterion we use for selecting the color space is based on the color distance uniformity, which means that the physical distance in the color space is proportional to the subjective perception distance. We have chosen the Lu*v* and La*b* color spaces as the basis for color representation. However, there are still drawbacks for using these uniform color spaces like Lu*v*. The main problem is that the translation process from RGB space to the Lu*v* is complicated and, more importantly, the reverse translation process is very difficult.

Usefulness of color-based features is still an open issue requiring more research. A technique based on color histograms is used in the QBIC system [24]. From the color histograms, other color-related features can be derived, such as average color, color intersection and color pairs. Color intersection was used in [23] to perform 3D object classification. Color pairs were used in [13] to capture the spatial correlations between adjacent color regions. Extension of color pairs was reported in [33] to reduce the effect of background images on the accuracy of the color pairs. In our system, we are taking two separate approaches — single color region extraction and quad-tree based color histogram indexing. We describe them separately in the following.

Single Color Region Extraction:

A simple but useful method using the image color features to capture the visual content is single color region extraction. First, we select a finite set of representative colors. For each representative color, we find the image

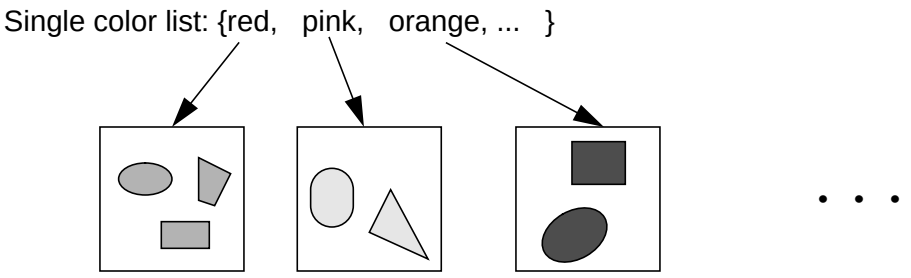


FIGURE 6. An inversion file approach for single color indexing.

regions whose pixels are within some small distance from the representative color. Once we have these color regions extracted, the file inversion approach of data indexing is used to store these single color regions. Figure 6 illustrates the single color data structure. Based on these single color regions, more sophisticated image queries can be formulated, such as those concerning the location, size, spatial relations of the color regions. Given the color key from users, scanning of the raw image data is no longer necessary. Instead, efficient search against the single color list can be utilized.

Several technical issues need to be resolved. Selection of the representative colors will determine the gamut range of the whole application. It needs to be optimized with joint consideration of computational efficiency. For specific applications, the representative color list can be obtained by training and clustering. Most typical colors can thus be measured and recorded. A general approach is to simply quantize a uniform color space such as the Lu*v* color space. For initial general testing, we have adopted the later approach.

Extraction of contiguous regions corresponding to each color can be done in different ways also. Each image in the database can be quantized against the representative color list, with appropriate distance functions defined in each color space. For example, a Euclidean distance measure may be sufficient in the uniform color space. This method will result in non-overlapping color regions in each image, with hard region boundaries among different colors. To take into account of fuzziness in our subjective perception, we adopted an alternative which achieves soft region boundaries at some cost of higher computational complexity. The computational process is as follows. For any given representative color, the distance between every pixel in the image and the sample color is calculated. The results are then stored in a distance image which has the same size as the original image. Non-linear rank filters are used to remove spot noise and improve region smoothness. The filtered distance image is then thresholded with some threshold level which can be set to be specific to each different color. A second threshold is also used to removed small regions at the end. Soft overlapped region boundaries are achieved by using adaptive distance thresholds for different colors. Also, this technique can be used independently of the choice of the color space and the distance function.

Quad-Tree Based Color Histogram Indexing:

Color histogram is supposed to capture much richer information than the single color approach. Color histograms are defined based on an image region. Other useful color information such as dominant color components and average colors can all be derived from the color histograms. However, once the raw image data is reduced to the color histogram, the spatial information is lost completely. To overcome this problem, we apply and extend the quad-tree based approach used in the texture indexing approach to the color histogram. Starting from the quad-tree spatial decomposition of an image, we measure the individual color histogram of each quad-tree terminal node, which corresponds to the smallest image unit. The size of the image unit should be determined based on the optimal tradeoff between the spatial resolution and color histogram reliability. Using an approach similar to the texture region segmentation, neighboring quad-tree nodes are compared and similar blocks are merged. One important issue is how to measure the color histogram distance.

The discussion of uniform color space which has uniform subjective color distance can not be directly extended to the color histogram. The Euclidean distance between two color histogram vectors is not suited in this case. In [24], a modified distance function was proposed to overcome this problem. The distance between two histograms $\grave{x}$ and $\grave{y}$ can be defined as

$$d(\vec{x},\vec{y}) = (\vec{x}-\vec{y})^{t} A (\vec{x}-\vec{y}) = \sum_{i,j} (x_i - y_i)\, a_{i,j}\, (x_j - y_j) \qquad \text{(EQ 2)}$$

where elements $a_{i,j}$ of matrix A represents the "cross correlation" between color i and color j. If A is the identity matrix, then (EQ 2) becomes the standard Euclidean distance. The compensation by $a_{i,j}$ coefficients is important because usually colors are not orthogonal. For example, the distance between red and orange should be smaller than that between red and blue. Therefore, $a_{i,j}$ between red and orange should be smaller than $a_{i,j}$ between red and blue.

Using color histograms to discriminate colors actually provides much flexibility. For example, the *average color* of an image region can be obtained easily by calculating the mean of the color histogram. In addition, as described in [23], *intersections of color histograms* can provide useful measurement of color similarity, such as "30% of the color distribution of image A is similar to the color distribution of image B". The color pair approach used in [13] was also based on the color histogram platform. However, none of the previous approaches explored the possibility of abstracting prominent regions based on the color histogram feature.

2.3 Shape

Shape feature is extremely useful in many image databases (such as electronics schematic matching) and pattern matching applications (such as military target recognition). Wavelet decomposition has been used to effectively detect edges in images. Mallat and Zhong used the derivative of Gaussian as the wavelet function and detect edge points at the maxima or zero-crossing points in different scales [26,25]. The same technique has been applied to signal approximation and image coding as well [26]. Related work of multi-scale edge detection has also been proposed by Canny in [27]. These prior efforts have provided an excellent framework for shape extraction in the wavelet domain, based on the assumption that shape information can be obtained by linking edge points in some smart ways. In [30], high-level reasoning techniques combining the specific domain knowledge were proposed to abstract object shapes from the underlying edge points. We think for practical applications the integration with domain knowledge is not only unavoidable, but also very beneficial.

In our study, we also combine the wavelet domain edge detection technique with the previously mentioned quad-tree based segmentation. There are two major advantages that the quad-tree-based technique can bring to edge detection in the wavelet domain. The first is for post-processing the edge information, in particular, "closing" and "tracing" scattered broken edges in different scales. Instead of globally tracing edge points with some smoothness constraints, a "split and link" approach based on the quad-tree decomposition can be used. Within each image block pointed by a quad-tree node, we perform local closing operation to produce connected edges. Then each edge is registered by its interception points with the block boundaries. To link edge segments from neighboring blocks, we compute the distance between interception points on the overlapping boundary. Edge segments from neighboring blocks are linked together if this distance is below some threshold. By using this "split and link" approach, edge closing operations are performed locally within an image block only, and thus the computational complexity can be reduced.

The second benefit of using the quad-tree based image decomposition is to *bind* the edge information with other features, such as texture and color, in the same data structure. We discuss this unexplored area in the next section.

3. An *Integrated Feature Map* for Content-Based Visual Query

The segmentation results from texture-based and color-based segmentation are *region-based*. In other words, each terminal node in the final tree structure represents an image region with homogeneous features (color or texture). The edge-based feature extraction produces closed or broken shapes. Integration of these final features in the same quad-tree structure of an image is very intriguing.

Figure 7 illustrates the concept of ***integrated feature map***. Regions produced from texture and color segmentations and shapes produced from edge detection are mapped onto the same quad-tree data structure. This can be easily implemented since we have used the same quad-tree structure in texture segmentation, color segmentation, and edge segment linking. There are many great potential applications of this *integrated feature map*.

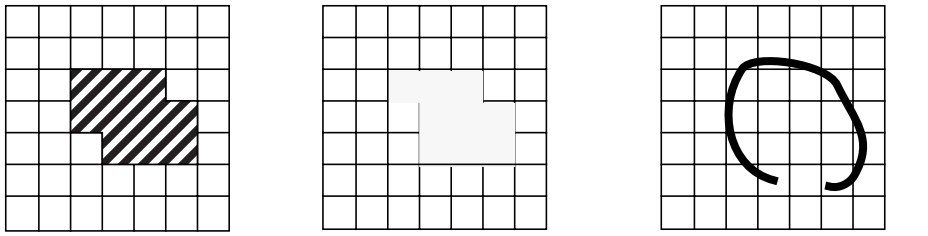


FIGURE 7. Integrating segmentation results based on different features (a) texture (b) color (c) edge/shape.

First, independent signal features can now be integrated to detect and index image regions/objects more effectively. Some objects may not be detectable by one single feature only. Consistency in segmentation results from different features will improve the confidence level of the object extraction results. Also, by incorporating domain-specific knowledge systems, we may be able to map the low-level feature maps to real-world objects in constraint applications. Indexing each image object by multiple features also provides higher flexibility in the query stage. For example, the same object can be searched by any or all of the associated features. A query example could be *"find images containing regions with this texture AND/OR this color pattern AND/OR this shape"*.

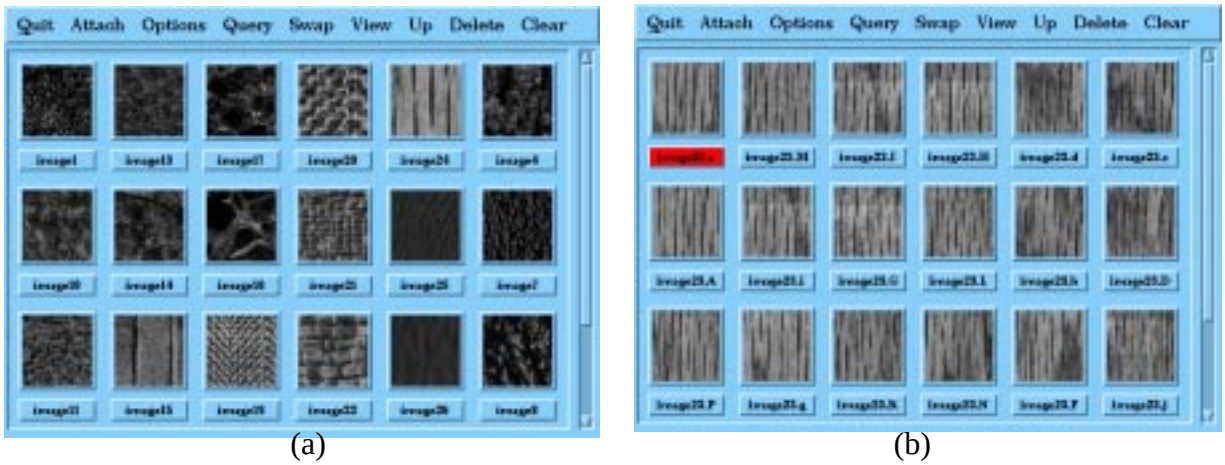
Secondly, boundary alignment between objects extracted by different features provides a new arena of image query. Some image regions may have well-aligned segmentation boundaries from different features. Some objects may not, depending on the physical appearance of the image objects. For example, with the proposed integrated feature map, the image database will allow users to issue queries like *"find image regions with this texture, this homogeneous color pattern in the center, but a random color distribution around the boundary"*, or *"find all image regions with homogeneous texture and well enclosed color patterns within the boundary (as opposite to, a color pattern spread over several texture regions)"*.

In the multi-scaled edge-extraction techniques, uncertainty still exists especially when the input image is noisy or there are strong interactions between close edges. Some shapes may remain broken after the closing operation. With the integrated feature map, this problem can be alleviated to some extent. *Inference* from segmentation results from other features, such as color and texture, can improve the edge accuracy and find the missing edge segment of an image object. For example, as shown in Figure 7, if there is indication of a large major object from other features, the missing edge segment can be supplemented by the boundary from the color/texture segmentation. More importantly, based on the unified quad-tree data structure of the image, it is easy to perform the above inference and find the image blocks where missing edge segments are supposed to be found or interpolated.

4. Prototype and Testbed

Part of our work on integrated investigation of image feature extraction, compression, and interactive manipulations has been incorporated into Columbia's Multimedia/Video-on-Demand Testbed, for which the first generation has been completed [17]. The testbed is intended to accommodate various multimedia-oriented research and development projects in the campus. Current large-scale projects undertaken include video-on-demand, distributed visual information systems, and medical image databases. Interoperability experiments, which allow interconnections with outside VOD testbeds and high-speed networks, have been initiated as well.

We have implemented prototypes for each individual visual feature. A Brodatz texture image database supporting texture-based discrimination is shown in Figure 8. Multi-resolution incremental retrieval of matched images is supported. Figure 9 shows the interactive system interface which allows users to navigate through different color spaces, to formulate desirable single color key, and to specify the color dissimilarity threshold levels in the single color retrieval mechanism. Formulation of color histogram queries can be supported by simple image cutting and editing tools. Currently, we are in the stage of integrating all these individual visual features to practical applications such as art image databases [34], medical image databases, and video-on-demand. For video indexing, we take a top-down approach to segment the whole image sequence into different scenes. Each video scene is assumed to have consistent visual feature content. Therefore, a representative image frame from each scene is sufficient to capture the rough con-



(a) (b)

FIGURE 8. Texture Discrimination System. (a) Interface Showing Texture Keys, (b) Results of Query-by-Texture using Brodatz texture 23 as the texture-key.

(a) (b)

FIGURE 9. (a) Color Space Navigator — a tool that allows users navigate through 3-D color space; the 6x6 color pane displays local space. (b) Color Agent — another tool that allows user to define single color query by using text description and multiple color space navigation.

tent for the entire scene. All the above signal features like texture, color, and shape can then be applied to index these representative image frames. In [35], we have reported a Video Indexing system supporting automatic scene change/ dissolve detection in the MPEG compressed domain.

5. Conclusions and Future Work

Visual feature based image query has opened a new area calling for advanced study of image understanding, image database indexing, and fast compressed image processing. In anticipation of future large visual information systems, exploration of maximum synergistic optimization between image feature extraction, image compression, and storage optimization needs to be pursued. In this paper, we describe one of our current efforts in using multi-dimen-

sional low-level signal features in image indexing and query. By stressing at the low-level features and using less stringent segmentation requirement, we hope to keep the indexing process automatic, without expensive user interventions. We also hope to minimize the implementation complexity by deriving signal features directly in the compressed domain, such as DCT, wavelet transform, and the MPEG domain. In addition, we proposed a modified quad-tree data structure for binding different signal features.

Technical barriers still exist in the way to fully automatic content extraction (ideally) in the compressed domains. One example is the difficulty in interpreting subjective color perception in the compressed domain. Unlike edge or texture which are more or less amenable to compression algorithms, color information is hard to capture in the compressed domain. Overall, the final goal is to capitalize on the visual content as much as possible in devising new compression algorithms supporting content-based access and manipulations. This objective actually is also more or less reflected in the spirit of the new video coding standard effort [29].

Acknowledgment:

We thank Louis Wang and Horace Meng for their contributions in the area of color histogram indexing and video scene change/dissolve detection.

6. References

1. S.-F. Chang and D.G. Messerschmitt, "Manipulation and Compositing of MC-DCT Compressed Video," IEEE Journal of Selected Areas in Communications, Special Issue on Intelligent Signal Processing, pp. 1-11, Vol. 13, No.1, Jan. 1995.
2. J.R. Smith and S.-F. Chang, "Quad-Tree Segmentation for Texture-Based Image Query" *Proceedings*, ACM 2nd Multimedia Conference, San Francisco, Oct. 1994.
3. J.R. Smith and S.-F. Chang, "Transform Features for Texture Classification and Discrimination in Large Image Databases," *Proceedings*, IEEE Intern. Conference on Image Processing, Austin, Nov. 1994.
4. T.L. Kunii, *Visual Database Systems*, Elsevier Science Publishers, 1989.
5. E. Knuth and L.M. Wegner, *Visual Database Systems II*, Elsevier Science Publishers, 1992.
6. P. Stanchev, A. Smeulders, and F. Groen, "An Approach to Image Indexing of Documents," in *Visual Database Systems II*, Elsevier Science Publishers, 1992.
7. R. Barber, W. Equitz, M. Flickner, W. Niblack, D. Petrovic, P. Yanker, "Efficient Query by Image Content for Very Large Image Database", COMPCON '93, San Francisco, CA., 1993, pp. 17-19.
8. E. Binaghi, I. Gagliardi, R. Schettini, "Indexing and Fuzzy Logic-Based Retrieval of Color Images", in Visual Database Systems II, Elsevier Science Publishers, 1992.
9. P. Brodatz, Textures: a Photographic Album for Artists and Designers, Dover, New York, 1965.
10. T. Chang and C.-C. J. Kuo, "Texture Analysis and Classification with Tree-Structured Wavelet Transform," IEEE Transactions on Image Processing, Vol. 2, No. 4, Oct., 1993.
11. N.S. Chang and K.S. Fu, "Query-by-pictorial-example," IEEE Transactions on Software Engineerings, Vol.SE-6, No.6, pp.519-24, Nov. 1980.
12. S.-K. Chang, Q.Y. Shi, and C.W. Yan, "Iconic Indexing by 2-D Strings," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-6, No. 4, pp.475-84, July 1984.
13. A. Nagasaka and Y. Tanaka, "Automatic Video Indexing and Full-Video Search for Object Appearances" In E. Knuth and L. M. Wegner, editors, *Video Database Systems, II*, Elsevier Science Publishers B.V., North-Holland, 1992, pp. 113 - 127.
14. C.K. Chui, *An Introduction to Wavelets*, Academic Press, San Diego, 1992.
15. I. Daubechies, *Ten Lectures on Wavelets*, CBMS-NSF Series in Applied Mathematics, SIAM, Philadelphia, 1992.
16. S.G.Mallat, "Multifrequency Channel Decompositions of Images and Wavelet Models," IEEE Transactions on ASSP, 37(12):2091-2110, 1989.

17. S.-F. Chang, D. Anastassiou, A. Eleftheriadis, J. Meng, S. Paek, S. Pejhan, and J.R. Smith, “Development of Advanced Image/Video Servers in the Video on Demand Testbed,” IEEE workshop on Visual Signal Processing and Communications, New Brunswick, NJ, Sep. 1994. (also in CU/CTR Technical Report 379-94-26)
18. S.W. Smoliar and H. Zhang, “Content-Based Video Indexing and Retrieval,” IEEE Multimedia Magazine, Vol.1, No.2, Summer 1994.
19. T. Caelli and D. Reye, “On the Classification of Image Regions by Color, Texture, and Shape,” Pattern Recognition, Vol. 26, No. 4, pp.461-470, 1993.
20. A. C. Bovick, “Analysis of Multiresolution Narrow-Band Filters for Image Texture Segmentation,” IEEE T. Signal Processing, Vol.39, No.9, Sept., 1991, pp.2025-43.
21. A.K. Jain and F. Farrokhia,” Unsupervised Texture Segmentation Using Gabor Filters,” Pattern Recognition, Vol. 24, No. 12, pp. 1167-86, 1991.
22. William R. Dillon and M. Goldstein, *Multivariate Analysis*, John Wiley & Sons, 1984.
23. M. Swain and D. Ballard, “Color Indexing,” International Journal of Computer Vision, &:1, pp. 11--32, 1991.
24. Wayne Niblack, et al., “The OBIC Project: Querying Images by Content Using Color, Texture, and Shape,” IBM RJ 9203 (81511), Feb, 1993.
25. S. Mallat, “Zero-Crossing of a Wavelet Transform,” IEEE Transactions on Information Theory, Vol. 37, No. 4, July 1991, pp.1019-33.
26. S. Mallat and S. Zhong, “Characterization of Signals from Multiscale Edges,” IEEE T-PAMI, Vol. 14, No. 7, July 1992, pp. 710-32.
27. J. Canny, “A Computational Approach to Edge Detection,” IEEE T-PAMI, Vol. PAMI-8, No. 6, Nov. 1986, 679-98.
28. J.R. Bach, A. Paul, and R. Jain, “A Visual Information Management System for the Interactive Retrieval of Faces,” IEEE Transactions on Knowledge and Data Engineering, Vol. 5, No. 4, Aug. 1993, pp.619-628.
29. MPEG-4 Call for Proposals, ISO/IEC JTC1/SC29/WG11 N0820, Nov. 1994.
30. Y. Lu and R.C. Jain, “Reasoning About Edges in Scale Space,” IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 14 No. 4, April 1992, pp. 450-468.
31. D.H. Ballard and C.M. Brown, *Computer Vision*, Prentice Hall, Inc., 1982.
32. Y.S. Hsu, S. Prum, J.H. Kagel, and A.C. Andrews, “Pattern Recognition Experiments in the Mandala/Cosine Domain,” IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-5, No. 5, pp. 521-9, Sep. 1983.
33. T.S. Chua, S.K. Lim, and H.K. Pung, “Content-Based Retrieval of Segmented Images,” Proceedings of ACM second Multimedia Conference, San Francisco, CA, 1994.
34. “Columbia Making Virtual Art Museum,” *Record*, Columbia University, Vol. 20, No. 19, March 3 1995.
35. J. Meng, Y. Juan and S.-F. Chang, “Scene Change Detection in a MPEG Compressed Video Sequence,” SPIE Symposium on Electronic Imaging— Digital Video Compression: Algorithms and Technologies, San Jose, Feb. 1995.