

COLOR IMAGE RETRIEVAL USING MULTISPECTRAL RANDOM FIELD TEXTURE MODELS

Alireza Khotanzad and Orlando J. Hernandez
 Department of Electrical Engineering
 Southern Methodist University
 Dallas, Texas 75275

ABSTRACT

This paper describes a system to query large color image databases to find similar images to a target image presented during the query action. The approach is to use color and texture information to index and match the images in the database. The texture information is obtained via modeling with the Multispectral Simultaneous Auto Regressive (MSAR) random field model. The color is represented by ratios of sample color means. The retrieval process involves segmenting the image into regions of uniform texture/color using an unsupervised histogram clustering approach that utilizes MSAR and color features. The current system is capable of retrieving images that contain same texture(s)/color to a query image. Work is in progress to develop more sophisticated retrieval algorithms based on metrics defined on the segmented regions.

1. INTRODUCTION

An ever-increasing usage of digital images and the wide acceptance of very large volume image databases give rise to the need for organizing them according to their content so they can be retrieved easily. Retrieval of image data based on pictorial content queries is an interesting and challenging problem actively worked on by the research community [1]. With the growth of multimedia computing and the spread of the Internet, more and more people have access to large databases and would have applications for such retrieval systems [1],[3],[4]

This paper describes a color texture-based retrieval system that finds images similar to a query image in a large database. The similarity criterion is based on the notion of containing same/similar color texture regions. The approach involves characterizing color texture using features derived from a class of multispectral random field models. These features are then used in an unsupervised histogram clustering-based segmentation algorithm to find regions of uniform texture in the query image. The retrieval process involves identifying those images in the database that contain the same kind of texture found in the query image.

2. Texture Characterization with Multispectral Simultaneous Autoregressive Model

The texture of the color images is characterized using a class of multispectral random field model called the Multispectral Simultaneous Autoregressive (MSAR) model [5], [6]. The MSAR model is shown to be effective for color texture synthesis and classification [5], [6]. For mathematical simplicity, the model is formulated using a toroidal lattice assumption. A location within a two-dimensional M by M

lattice is denoted by \mathbf{s} . The value of an image observation at location \mathbf{s} is denoted by the vector value $\mathbf{y}(\mathbf{s})$, and the image

observations are assumed to have zero mean. The MSAR model relates each lattice position to its neighboring pixels, both within and between image planes, according to the following model equation:

$$y_i(\mathbf{s}) = \sum_{j=1}^P \sum_{\mathbf{r} \in N_{ij}} \mathbf{q}_{ij}(\mathbf{r}) y_j(\mathbf{s} \oplus \mathbf{r}) + \sqrt{\rho_i} w_i(\mathbf{s}), \quad i = 1, \dots, P$$

$y_i(\mathbf{s}) = i^{\text{th}}$ element of a zero mean observation vector

\mathbf{s} and $\mathbf{r} =$ two dimensional lattices

$P =$ number of image planes (in this case, $P = 3$, for the three image color planes: Red, Green, and Blue)

$N_{ij} =$ neighbor set relating pixels in plane i to neighbors in plane j (only interplane neighbor sets, i.e. $N_{ij}, i \neq j$, may include the (0,0) neighbor)

$\theta_{ij} =$ coefficients which define the dependence of $y_i(\mathbf{s})$ on the pixels in its neighbor set N_{ij}

$\rho_i =$ noise variance of image plane i

$w_i(\mathbf{s}) =$ i.i.d. random variables with zero mean and unit variance

\oplus denotes modulo M addition in each index

The parameters associated with the MSAR model are \mathbf{q} and \mathbf{r} vectors which collectively characterize the spatial interaction between neighboring pixels. These vectors are taken as features representing the underlying color texture present in the M by M image.

3. Color Characterization

In addition to modeling color texture, the color content of the image by itself is also important. Additional features focusing on the color alone are also considered. This is done

using the sample mean of the three color components in the red, green, and blue (RGB) space. The defined feature vector is:

$$\mathbf{f}_R = \left\{ \frac{\hat{m}_r}{\hat{m}_g}, \frac{\hat{m}_r}{\hat{m}_b} \right\}$$

with \hat{m} s being the sample mean of the respective color component. This is a two-dimensional feature vector containing ratios of the color means. It is assumed that the observed value at each pixel is a product of illumination and spectral reflectance. Under this assumption, the ratios of the color means are invariant to uniform changes in illumination intensity, i.e. the power of the illumination source changes uniformly across the spectrum. Such a change in illumination would result in changing each \hat{m} by a scale factor making the ratios invariant to illumination changes. This property makes the color features more robust

4. Segmentation with a Histogram-Based Clustering Algorithm

The first step in the retrieval process is to segment the query image into regions of uniform color texture. It should be noted that since the ultimate goal is to retrieve images similar to the one presented to the system, it suffices to find dominant regions of texture in the image and locating very exact boundaries of regions is not as important.

The segmentation algorithm used in this work relies on scanning the image with a sliding window and extracting texture and color features from each window. These features are then clustered using an unsupervised histogram-based algorithm. Mapping the identified clusters back into the image domain results in the desired segmentation

4.1 Feature Extraction with a Sliding Window

The windowing operation consists of sliding a window from left to right and top to bottom across the image as illustrated in Figure 1. M is the size of the image in pixels, W is the size of the window in pixels, and D is the size of the sliding step in pixels. D is set to 4 pixels for this work. To find the optimum window size for each case, the size of the window W varies from $W_1 = 4$ pixels to $W_2 = 28$ pixels in increments of 4 pixels. In later sections, it is explained how the best window size is decided.

The texture bounded by each window is characterized using the MSAR and color features. The neighborhood used for the MSAR model is a set that contains neighbors above, below, to the left, and to the right of the pixel as illustrated in Figure 2. This neighbor set is used for both inter and intra-planes of the model.

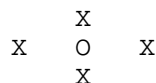


Figure 2. Neighbor Set used with the MSAR model

This neighbor set results in a 20-dimensional MSAR feature vector. Therefore together with the color feature set, a 22-

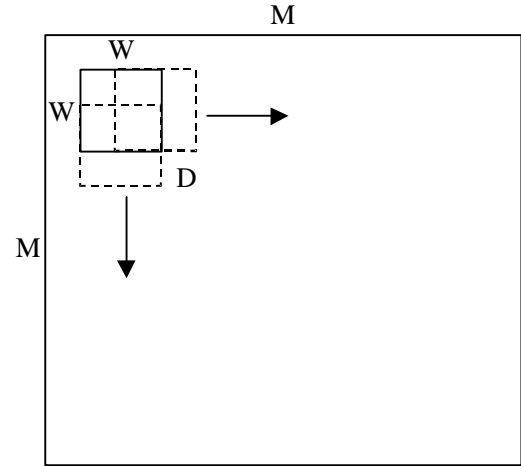


Figure 1. Feature Vector Extraction with a Sliding Window.

dimensional feature vector is used to characterize each window.

4.2 Clustering Algorithm

Once all 22-dimensional features are extracted from the sliding window, they are clustered in the feature space using an unsupervised histogram-based peak climbing algorithm [7], [8], [9]. The 22-dimensional histogram is generated by quantizing each dimension and dividing the space into hyperboxes. Next, the number of feature vectors falling in each hyperbox is counted and this count is associated with the respective hyperbox creating the required histogram.

After the histogram is generated in the feature space, a peak climbing clustering approach is utilized to group the features into distinct clusters. This is done by locating the peaks of the histogram. In Figure 3 this peak climbing approach is illustrated for a two-dimensional space.

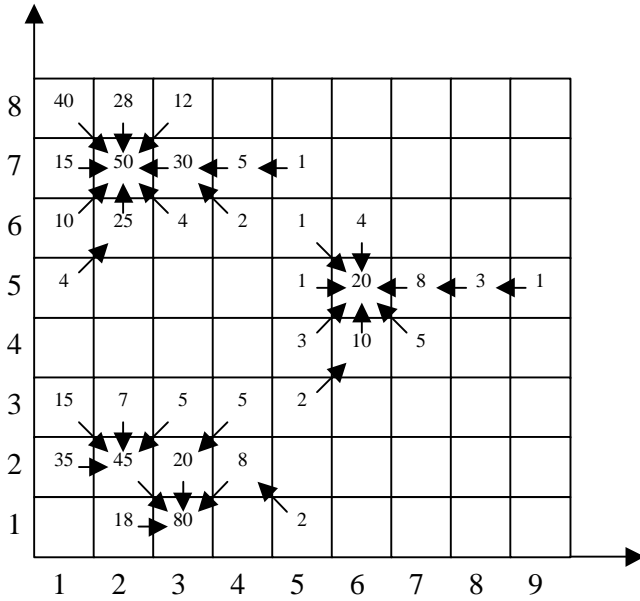


Figure 3. Illustration of the Peak Climbing Approach for a Two-dimensional Space.

The number in each cell (hyperbox) represents a hypothetical count for the feature vectors captured by that cell. By examining the counts of the 8-neighbors of a particular cell, a link is established between that cell and the closest cell having the largest count in the neighborhood. At the end of the link assignment, each cell is linked to one parent cell, but can be parent of more than one cell. A peak is defined as being a cell with the largest density in the neighborhood, i.e. a cell with no parent. A peak and all the cells that are linked to it are taken as a distinct cluster representing a mode in the histogram. Once the clusters are found, the windows associated with features grouped in the same cluster are tagged as belonging to the same category.

A major component of this algorithm is the number of quantization levels associated with each dimension. To decide this parameter, the total number of non-empty cells and the percentage of them capturing only one vector for each selection of quantization levels are examined. The best number of quantization levels is selected as the largest one that maximizes the measure below [9].

$$M_i = (N_{ci} - N_{ui}) \times N_{ui}$$

N_{ci} = number of non-empty cells

N_{ui} = number of cells capturing only one sample

The algorithm also includes a spatial domain cluster validation step. This step involves constructing a matrix B for each cluster m as:

$$B_m(i, j) = 1 \quad \text{if sample} \in \text{cluster } m$$

$$B_m(i, j) = 0 \quad \text{otherwise}$$

The (i,j) index corresponds to the location of a window used to scan the image. A cluster is considered compact if

only a very small number of its elements (1s) have a 0 neighbor, i.e. a cluster is considered valid (compact) if only a very small number of its elements have neighboring elements that do not belong to that cluster. A cluster that does not pass this test is merged with a valid cluster that has the closest centroid to it.

During the segmentation process, the best window size to scan the image with is chosen in an unsupervised fashion. The optimum window size is obtained by sweeping the image with varying window sizes, and choosing the smallest one out of at least two consecutive window sizes that produce the same number of clusters. The parameters for this operation are those described in Section 4.1.

4.3 Segmentation Results

The performance of the proposed segmentation algorithm and the associated features is illustrated in Figure 4. This figure shows six images each containing a number of different textures. These image mosaics are created from texture samples available in [10]. Below each image the segmentation result is presented in the form of a gray-level image with pixels belonging to the same texture having the same gray-level. At the top of the image, the size of the optimal window found by the algorithm is also shown. It is observed that the proposed algorithm performs quite well and is capable of finding the number and location of uniform textures in each image.

5. IMAGE RETRIEVAL PROCEDURE

The retrieval system is based on the described segmentation approach and the associated features. All the images in the database are first segmented. The texture associated with each region of each image is then characterized by a single 22-dimensional feature vector. This is done by fitting the maximum fitting square to each region and extracting features from this square. Of course this operation could be done in an off-line manner to speed up the process during the query phase.

When a query image is presented, a similar procedure is performed. For instance, the image may be segmented into three regions. A feature vector associated with each of the regions is then extracted from the corresponding maximum fitting square. Next, these three feature vectors are presented to the space containing the feature vectors of the database one at a time. For each case, those images that contain a feature vector similar to the one being presented are found. This is done with a simple clustering algorithm involving minimization of Euclidean distance. These images are then retrieved from the database.

It should be noted that the current approach only retrieves images that contain a textured region similar to one of the regions in the query image. Our on-going work is on advancing this algorithm to take into account the existence of

multiple regions within the image, their relative position and size, and the closeness of the retrieved images with respect to these measures.

5.1 Retrieval Results

Two examples of the performance of the retrieval system are shown in Figures 5 and 6. Figure 5 involves synthetic textures whereas a real scene is considered in Figure 6. In both figures the top left image with blue border is the query image. Seven images retrieved from the considered databases are shown next to the query image in no particular order. The number of images in the databases are 102 and 51 for the synthetic and real images, respectively. Note that in case of synthetic textures, all the retrieved images have at least one texture region in common with the query image. The real scene case pulls images that are visually close to the query image.

6. CONCLUSIONS

This paper presents a color image retrieval system that is based on modeling color textures in the image. The texture is characterized with a multispectral random field model. Features associated with this model along with color-only features are used to segment the image. A histogram-based clustering algorithm is used for this purpose. Once the image is segmented, features associated with each of its regions are used to find images similar to it in a database. Experimental results with synthetic and natural images are presented.

7. REFERENCES

[1] B. S. Manjunath and W. Y. Ma, "Texture Features for Browsing and Retrieval of Image Data," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, August 1996, pp. 837-842.

[2] M. Shneier and M. Abdel-Mottaleb, "Exploiting the JPEG Compression Scheme for Image Retrieval," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, August 1996, pp. 849-853.

[3] G. Healey and A. Jain, "Retrieving Multispectral Satellite Images Using Physics-Based Invariant Representations," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, August 1996, pp. 842-848.

[4] S. A. Chien and H. B. Mortensen, "Automating Image Processing for Scientific Data Analysis of a Large Image Database," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, August 1996, pp. 854-859.

[5] J. W. Bennett and A. Khotanzad, "Multispectral Random Field Models for Synthesis and Analysis of Color

Images," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, March 1998, pp. 327-332.

[6] J. W. Bennett, "Modeling and Analysis of Gray Tone, Color, and Multispectral Texture Images by Random Field Models and Their Generalizations," *Southern Methodist University*, Dissertation, May 1997.

[7] Khotanzad and J. Y. Chen, "Unsupervised Segmentation of Textured Images by Edge Detection in Multidimensional Features," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 11, no. 4, April 1989, pp. 414-421.

[8] J. Y. Chen, "Segmentation of Textured Images by Edge Detection in Multidimensional Features," *Southern Methodist University*, Dissertation, September 1987.

[9] Khotanzad and A. Bouarfa, "Image Segmentation by a Parallel, Non-Parametric Histogram Based Clustering Algorithm," *Pattern Recognition*, vol. 23, no. 9, September 1990, pp. 961-963.

[10] Vision and Modeling Group, MIT Media Laboratory, "Vision Texture (VisTex) database," <http://www-white.media.mit.edu/vismod/>, 1995.

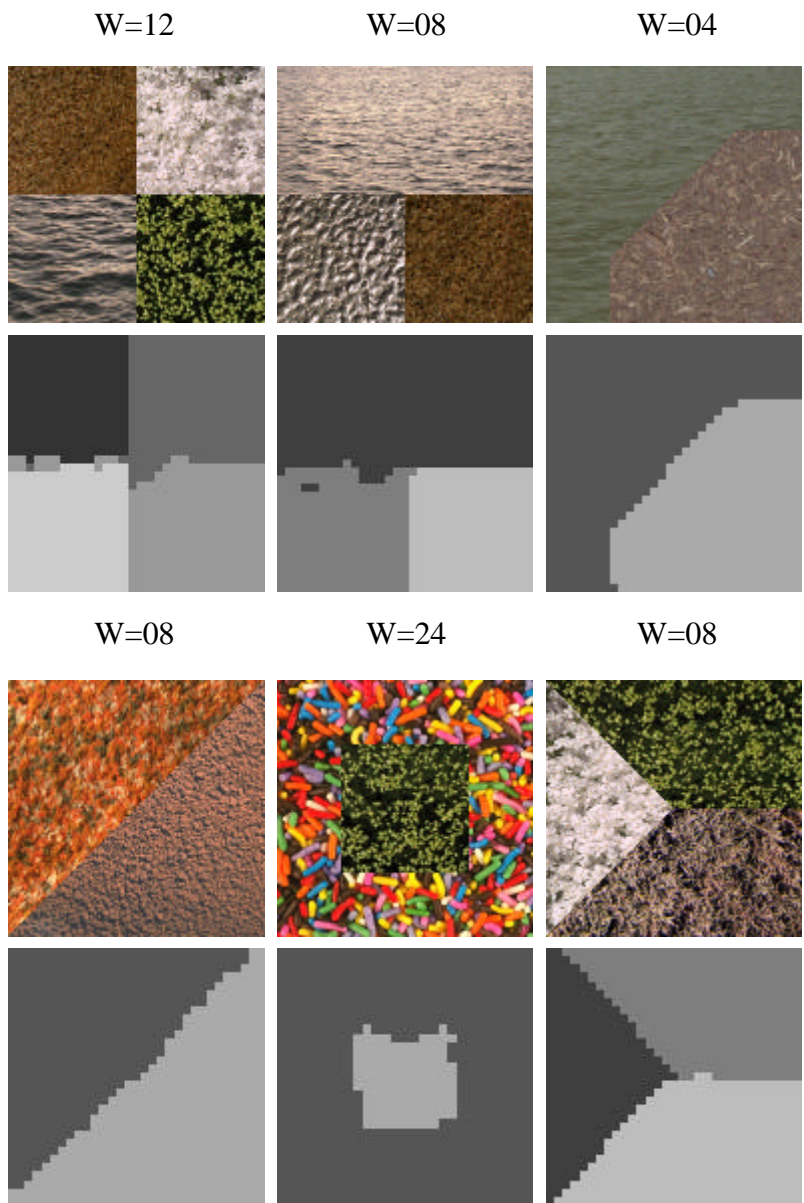


Figure 4. Segmentation results for texture image mosaics. The optimum window size found by the algorithm is shown at the top of each image.

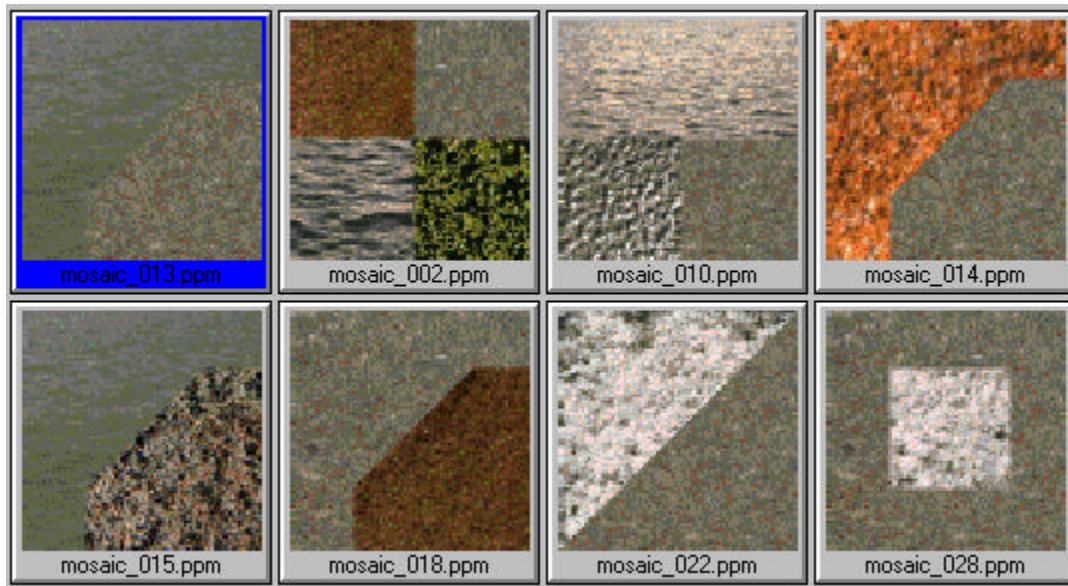


Figure 5. Retrieval results for the query image at top left



Figure 6. Retrieval results for the natural scene query image at top left.