Combined Low-Level Descriptors for Improving the Retrieval Performance

MOHAMED EISA
Computer Science Department
Mansoura University
35516 Mansoura, Egypt
EGYPT

Abstract: - Most of the current visual descriptors used are calculated for full images. The local image areas of interest are easily left unnoticed as the global features do not contain enough information for local discrimination. The main contributions of this paper is enhancing the matching performance by applying different kinds of visual descriptors (color, Texture, Edge) to the sub-image areas without using any type of segmentation and compare the obtained feature descriptors separately. Three feature extraction methods, which are block-based descriptors, are presented. The first one is an advanced color feature extraction derived from the modification of Stricker's method. The second one is a texture feature extraction using the Local Binary Pattern (LBP) which is invariant, fast to calculate and its efficiency originates from the detection of different micro patterns. The third one is an edge feature extraction using the Edge Histogram Descriptor (EHD) which is time-consuming as well as computationally expensive. The experimental results demonstrate that block-based feature descriptors have good performance in terms of matching efficiency and effectiveness.

Keywords: - MPEG-7, Visual Descriptors, Similarity Measure, Edge Histogram, Texture and Color Features.

1 Introduction
The growing number of image and video databases in the internet and other information sources has forced researchers to strive after better retrieval methods. There is certainly a continuous need for novel ideas in all areas of Content-Based Image Retrieval (CBIR) [1]. The most common categories of feature descriptors are based on color, texture and shape and there are many alternatives in each of these [2].

Most of the current CBIR texture descriptors used in commercial systems is calculated for full images. The full image approach is well justified as it usually keeps the size of the feature database reasonably low, depending on the used features and the amount of images. Still there is a problem while considering only full images. The local image areas of interest are easily left unnoticed as the global features do not contain enough information for local discrimination [3]. A way to pay attention to local properties is to use image segmentation. However, the segmentation is usually prone to errors so it is not very suitable for images with general content. Another way to enhance the retrieval results is to apply the image extractor to the sub-image areas without using any type of segmentation and compare the obtained feature descriptors separately.

In general, histogram-based methods provide overall image characteristics. However, they show high sensitivity to illumination and can not represent localized features well. An advanced color extraction method to compensate the above drawbacks of histogram-based methods is proposed. The most important property of the LBP operator in real world applications is its tolerance against illumination changes. Another important is its computational simplicity, which makes it possible to analyze images in challenging real-time settings [4, 19].

Edges in images constitute an important feature to represent their content. An edge histogram in the image space represents the frequency and the directionality of the brightness changes in the image. It is a unique feature for images, which can not be duplicated by a color histogram or the texture features [5, 20]. The edge histogram descriptor expresses only the local edge distribution in the image which may not be sufficient to represent global features of the edge distribution. So the semi-global and global edge histograms from the local histogram bins are presented.
The rest of this paper is composed of the following sections. In section 2. The visual descriptors depending on dividing the image into sub-block are introduced. The similarity matching for each feature is presented separately and also for combined features in section 3. Experimental results are shown in section 4. Finally, the conclusion is presented in section 5.

2 Visual Descriptors

2.1 Color Feature Extraction

There have been researches using color feature as a query key for image retrieval in general, they use color histogram methods to obtain color feature information from an image. Color histogram indexing was introduced by Swain et al. [6]. They used the brightness values of RGB color space and stored coarsely quantized color histogram of images as indices. Funt et al. modified Swain’s method to improve its performance by using the histogram of the RGB color ratio from neighboring locations [7]. In general, histogram-based methods provide overall image characteristics. However, they show high sensitivity to illumination and can not represent localized features well. An advanced color extraction method to compensate the above drawbacks of histogram-based methods is proposed. The block diagram of the color feature extraction is shown in Fig.1.

First, when a query image is presented its RGB color space is converted into HSI (Hue, saturation, and intensity) for better retrieval performance. Next, each image is divided into sub-blocks for considering spatial location information. Then, we can get a color feature vector from each sub-block. Using the color feature vectors and the measure of similarity mentioned in section 3, we can retrieve similar images from an image database with color feature only.

2.1.1 The Color Model Conversion and Sub-block Division

Each RGB input image is converted into a HSI image using the following:

\[
H = \cos^{-1}\left(\frac{1}{2} \left[ \frac{(R-G)+(R-B)}{(R-G)^2 + (R-B)(G-B)} \right] \right),
\]

\[
S = 1 - \frac{3}{(R+B+G)} \min(R,G,B),
\]

\[
I = \frac{1}{3}(R+G+B). \tag{1}
\]

To obtain spatial localization information in the input image, it is divided into \(m \times n\) size sub-blocks.

2.1.2 Color Feature Vector Extraction

Stricker proposed a color extraction method using the three moments of each color channel of an image: average, standard deviation and skewness [8]. For the feature vector extraction, its color features are computed by the following equations:

\[
E_i = \frac{1}{m \cdot n} \sum_{j=1}^{m \cdot n} P_{ij}, \quad \sigma_i = \left[ \frac{1}{m \cdot n} \sum_{j=1}^{m \cdot n} (P_{ij} - E_i)^2 \right]^{1/2},
\]

and

\[
\alpha_i = \left[ \frac{1}{m \cdot n} \sum_{j=1}^{m \cdot n} (P_{ij} - E_i)^3 \right]^{1/3} \tag{2}
\]

Where \(E_i\) is an average of each color channel \((i = H, S, I)\), \(\sigma_i\) is a standard deviation, and \(\alpha_i\) is skewness. \(P_{ij}\) is the value of each color channel at the \(j\)-th image pixel. \(m \cdot n\) is a total number of pixels per image. Stricker used the 9 feature vector which consists of three moments for each color channel, H, S, I, respectively. The histograms are quantized to 16 bins for H channel and 4 bins for S and I channels respectively.

A modification to the color extraction algorithm proposed by Stricker is proposed, to improve performance as follows:
• to use the color feature vectors considering spatial location information, Stricker's method to each sub-block is applied;
• the number of bins for the index in S and I channel is reduced to 2 while keeping the 16 bins for index in H channel, because S and I do not much affect the retrieval performance directly,
• Only, the first moment for S and I channels except for H is used, for which also, the second and third moments are calculated. Therefore, 5 color features are used instead of Stricker's 9 features.

For the k-th \( m \cdot n \) sub-block, we use the 5 features vector: The average \( (E_{k,H}) \), standard deviation \( (\sigma_{k,H}) \), and skewness \( (\alpha_{k,H}) \) for H channel, and the average \( (E_{k,S}, E_{k,I}) \) for S and I channels, respectively.

2.2 Texture Descriptor

The original LBP method [9], shown in Fig. 2, was first introduced as a complementary measure for local image contrast. It operated with eight neighboring pixels using the center as a threshold. The final LBP code was then produced by multiplying the thresholded values by weights given by powers of two and adding the results in a way described by Fig. 2. By definition, LBP is invariant to any monotonic transformation of the gray scale and it is quick to compute.

The original LBP has been extended to a more generalized approach [10] in which the number of neighboring sample points is not limited. In Fig. 3, three different LBP operators are given. The predicate (radius, \( R \)) has no constraints like in the original version and the samples (\( P \)) that do not fall on exact pixel positions are interpolated by using bilinear interpolation.

With larger neighborhoods, the number of possible LBP codes increases exponentially. This can be avoided, to some extent, by considering only a subset of the codes. One approach is to use so called uniform patterns [10] representing the statistically most common LBP codes. With them the size of the feature histogram generated by an LBP operator can be reduced without significant loss in its discrimination capability. For example, if we consider only those LBP codes that have U value of 2 (U refers to the measure of uniformity, that is the number of 0/1 and 1/0 transitions in the circular binary code pattern), in case of a \( 3 \times 3 \) \( (LBP_{3\times3}^{8,1}) \) operator we get a feature vector of 58 bins instead of original 256 bins. When the remaining patterns are accumulated to a single bin the histogram becomes 59. That is only a fraction \( (59/256) \) of the original. The spatial support area of LBP feature extractor can be extended by using operators with different radii and sample counts and combining the results [10]. By utilizing \( N \) operators we get \( N \) different LBP codes which can be connected to form a single feature descriptor vector of \( N \) codes. While inserting the marginal distributions of feature extractors one after another, the distance between the sample and model is given by Eq. 3:

\[
L_N = \sum_{n=1}^{N} L(S^n, M^n),
\]

where \( S^n \) and \( M^n \) are the sample and model distributions extracted by the \( n \)th operator.

![Original LBP](image1)

![General LBP](image2)

2.2.1 The Block Division Method

The block division method is a simple approach that relies on sub-images to address the spatial properties of images. It can be used together with any histogram descriptors similar to LBP. The method works in the following way: First it divides the model images into square blocks that are arbitrary in size and overlap. Then the method calculates the LBP distributions for each of the blocks and combines the histograms into a single vector of sub-histograms representing the image. In the query phase the same is done for the query image(s) after which the query and model are compared by
calculating the distance between each sub-histogram of the query and model. The final image dissimilarity $D$ for classification is the sum of minimum distances as presented by Eq. 4:

$$D = \sum_{i=0}^{N-1} \min_j \left( D_{i,j} \right)$$

(4)

Where $N$ is the total amount of query image blocks and $D_{i,j}$ the distance (relative $L_1$, as shown in section 3) between the $i$th query and $j$th model block histograms. An example of the approach in operation is shown in Fig. 4. Note that, in this figure the shown histograms are only examples and not the actual LBP distributions of corresponding image blocks.

Table 1 summarizes the semantics of the 80-bin EHD [11, 12].

<table>
<thead>
<tr>
<th>Histogram bins</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>BinCounts[0]</td>
<td>Vertical edge of sub-image at (0,0)</td>
</tr>
<tr>
<td>BinCounts[1]</td>
<td>Horizontal edge of sub-image at (0,0)</td>
</tr>
<tr>
<td>BinCounts[2]</td>
<td>45-degree edge of sub-image at (0,0)</td>
</tr>
<tr>
<td>BinCounts[3]</td>
<td>135-degree edge of sub-image at (0,0)</td>
</tr>
<tr>
<td>BinCounts[4]</td>
<td>Nondirectional edge of sub-image at (0,0)</td>
</tr>
<tr>
<td>BinCounts[5]</td>
<td>Vertical edge of sub-image at (0,1)</td>
</tr>
<tr>
<td>BinCounts[78]</td>
<td>135-degree edge of sub-image at (3,3)</td>
</tr>
<tr>
<td>BinCounts[79]</td>
<td>Nondirectional edge of sub-image at (3,3)</td>
</tr>
</tbody>
</table>

Fig. 5. Definition of sub-image and image-blocks

2.3 MPEG-7 Edge Histogram Descriptor

2.3.1 Definition and Semantics
Spatial distribution of edges in an image is another useful texture descriptor for similarity search and retrieval. The EHD in MPEG-7 represents local edge distribution in the image. Specifically, dividing the image space into $4 \times 4$ sub-images as shown in Fig. 5, the local-edge distribution for each sub-image can be represented by a histogram. To generate the histogram, edges in the sub-images are categorized into five types; vertical, horizontal, 45-degree diagonal, 135-degree diagonal and nondirectional edges. Since there are 16 sub-images, a total of $5 \times 16 = 80$ histogram bins are required.
the corresponding edge orientation is associated with the image-block. If the maximum of the edge strengths is below the given threshold, then that block is not classified as an edge block. The global edge histogram and some semi-global edge histograms are generated directly from BinCounts[i], i=0,…,79. The global histogram represents the edge distribution for the whole image space. Since there are 5 edge types, the global edge histogram has 5 bins and each bin value is obtained by accumulating and normalizing the dequantized bin values of the corresponding edge type of BinCounts[]. Similarly, for the semi-global edge histograms, we can group some subsets of BinCounts[].

13 different segments are defined and for each segment, edge distributions for five different edge types from the 80 local histogram bins are generated. Consequently, we have a total of 150 bins (80 bins (local) + 5 bins (global) + 65 bins (13×5, semi-global)) for the similarity matching.

![Fig.6. Filter coefficients for edge detection](image)

3. Similarity matching

For the similarity calculation, to compare the color feature vectors of the query image Q and a target image C in the database, we used

$$D(Q,C) = \sum_{k=1}^{l} \left| F_{q,k} - F_{c,k} \right|$$

(5)

Where \(F_{q,k}\) is the color feature vector of the k-th sub-block in image Q and \(F_{c,k}\) is that of image C, \(l\) is the number of sub-blocks [13].

To compare the texture feature histograms, a relative \(L_t\) measure was chosen due to its performance in terms of both speed and good retrieval rates.

$$L_t^{relative}(Q,C) = \sum_{k=1}^{l} \left| \frac{L_{q,k} - L_{c,k}}{L_{q,k} + L_{c,k}} \right|$$

(6)

Where \(L_q\) and \(L_c\) represent the texture feature histograms to be compared and \(k\) is the corresponding bin of the sub-block in image [14].

Comparing the edge feature histograms requires the distance \(D(Q,C)\) of two image histograms \(Q\) and \(C\) using the following measure:

$$D(Q,C) = \sum_{i=0}^{79} \left| Local_{Q[i]} - Local_{C[i]} \right| + 5 \sum_{i=0}^{4} \left| Global_{Q[i]} - Global_{C[i]} \right| + \sum_{i=0}^{64} \left| Semi\_Global_{Q[i]} - Semi\_Global_{C[i]} \right|$$

(7)

Where \(Local_{Q[i]}\) represents the reconstructed value of Bin_Count[i] of image Q and \(Local_{C[i]}\) is that of image C [5], \(Global_{Q[i]}\) and \(Global_{C[i]}\) represent the normalized bin values for the global edge histograms of image Q and C, respectively. Similarly, \(Semi\_Global_{Q[i]}\) and \(Semi\_Global_{C[i]}\) represent the normalized histogram bin values for the semi-global edge histograms of image Q and C, respectively. Since the number of bins of the global histogram is relatively smaller than that of local and semi-global histograms, a weighting factor of 5 is applied in (7).

Let \(D_e\) be the result of the difference between the query image and a database based on color feature vectors, \(D_t\) is the result of the difference between the query image and a database based on texture histogram and \(D_c\) is the result of the difference between the query image and a database based on edge histogram, then the total difference take the form [15, 16]:

$$D_{cte} = \alpha D_c + \beta D_t + \gamma D_e,$$

where \(\alpha + \beta + \gamma = 1\)

(8)

4. Experimental Results

Experiments were conducted using an image database consisting of Commercial Corel Image Gallery [17] images of size \(384 \times 256\). Six images selected in the database were used to make query images and ground-truth images (the ground-truth data is a set of visually similar images for a given query image) are categorized by image classes based on the query images as shows in Fig. 7.

![Fig.6. Filter coefficients for edge detection](image)
The ground-truths that used in experiments were determined by three participants of the MPHG-7 CE. As a measure of retrieval accuracy, the Average Normalized Modified Retrieval Rank (ANMRR) [18] is used. Precision and Recall are well known measures for the retrieval performance. They are basically a "hit and miss" counter. That is, the performance is based on the number of retrieved images, which have similarity measures that are greater than a given threshold. For more specific comparisons, the rank information among the retrieved images is needed. ANMRR is the measure that exploits the rank of the retrieved images as well. Note that lower ANMRR values indicate more accurate retrieval performance.

Table 2 shows the performance comparisons of NMRR and ANMRR between the modification of Sticker's method, LBP, EHD and the proposed method.

<table>
<thead>
<tr>
<th>NMRR</th>
<th>M. of S. method</th>
<th>LBP</th>
<th>EHD</th>
<th>Proposed method</th>
</tr>
</thead>
<tbody>
<tr>
<td>query1</td>
<td>0.7823</td>
<td>0.7503</td>
<td>0.7235</td>
<td>0.7024</td>
</tr>
<tr>
<td>query2</td>
<td>0.8326</td>
<td>0.8153</td>
<td>0.8410</td>
<td>0.7803</td>
</tr>
<tr>
<td>query3</td>
<td>0.9024</td>
<td>0.7904</td>
<td>0.8404</td>
<td>0.8105</td>
</tr>
<tr>
<td>query4</td>
<td>0.7323</td>
<td>0.7158</td>
<td>0.7987</td>
<td>0.6835</td>
</tr>
<tr>
<td>query5</td>
<td>0.4685</td>
<td>0.4375</td>
<td>0.4643</td>
<td>0.4185</td>
</tr>
<tr>
<td>query6</td>
<td>0.6204</td>
<td>0.6185</td>
<td>0.6543</td>
<td>0.5854</td>
</tr>
<tr>
<td>ANMRR</td>
<td>0.7231</td>
<td>0.6880</td>
<td>0.7204</td>
<td>0.6635</td>
</tr>
</tbody>
</table>
Figs. 8-11 demonstrate retrieval results for some query images. As you can see, the proposed method retrieves semantically more similar images. In Figs. 8-11, the left-upper image is a query and its first ranked image. The other images are displayed in a raster scan order according to the retrieval ranks.

5. Conclusion

Visual descriptors for increasing the matching performance were presented. Three feature extraction methods based on color, texture and edges were proposed.

The first one is an advanced color feature extraction algorithm considering spatial location information.

The second one is a texture feature extraction using LBP with block division method that relies on sub-images to address the spatial properties of images. The third one is an edge feature extraction using EHD that is very flexible; it consists of local edge histograms which using them to generate various patterns of semi-global edge histograms and a global edge histogram.

It must be emphasized that the main objective of this paper is increasing the retrieval performance, not only by using the visual descriptors separately but also by using the all mentioned visual descriptors to extract the all possible features from the same image. Consequently, we can deal with different kind of images that have color, texture and edges in the same time.

References:


