

# Compact color descriptor for fast image and video segment retrieval<sup>1</sup>

Santhana Krishnamachari and Mohamed Abdel-Mottaleb

Philips Research  
345 Scarborough Road  
Briarcliff Manor, New York 10562  
{sgk,msa}@philabs.research.philips.com

## ABSTRACT

Histograms are the most prevalently used representation for the color content of images and video. An elaborate representation of the histograms requires specifying the color centers of the histogram bins and the count of the number of image pixels with that color. Such an elaborate representation, though expressive, may not be necessary for some tasks in image search, filtering and retrieval. A *qualitative* representation of the histogram is sufficient for many applications. Such a representation will be compact and greatly simplify the storage and transmission of the image representation. It will also reduce the computational complexity of search and filtering algorithms without adversely affecting the quality. We present such a compact binary descriptor for color representation. This descriptor is the quantized Haar transform coefficients of the color histograms. We show the use of this descriptor for fast retrieval of similar images and search for similar video segments from a large database. We also show the use of this descriptor for browsing large image databases without the need for computationally expensive clustering algorithms. The compact nature of the descriptor and the associated simple similarity measure allows searching over a database of about four hours of video in less than 5-6 seconds without the use of any sophisticated indexing scheme.

This compact descriptor is currently being evaluated for inclusion in MPEG-7, the upcoming ISO standard for describing multimedia content.

Keywords: Compact descriptor, binary histogram, image retrieval, video segment matching, browsing

## 1. INTRODUCTION

Content-based search and retrieval techniques for multimedia content utilize various descriptors for different image features. The color content, being one of the easily computable features is often used in content-based search<sup>1,2</sup> and filtering<sup>3</sup> applications. Color histograms are the most commonly used representation of the color content and can describe the color content of an image very well. However, in general, they need relatively large number of bits for representation. Using such a representation may not pose a problem for applications with a large storage capacity and with ample computational resources. However for applications like content filtering, where the multimedia data and related descriptions are processed by a set-top box or a similar device with limited computational resource and limited bandwidth, it is imperative to use a compact representation. Even for a database application, histograms with a large number of bins pose difficult problems in indexing.

Figure 1 shows a pyramidal structure of various color descriptors based on their size. At the bottom of the pyramid are the detailed color descriptors, such as a color histogram. The size of these color descriptors is fairly large, about 256 bytes, and they offer an excellent performance in terms of color-based search and filtering. These descriptors are useful for applications that have large storage capacity, sophisticated indexing schemes, large computational resource and no bandwidth restrictions. At the top of the pyramid are the compact color descriptors; the sizes of these descriptors is very small, typically less than 128 bits. These descriptors offer a very good performance, although arguably not as good as the detailed descriptors. These descriptors are useful applications where the computational resources are limited and there is a need to stream the descriptors through a bandwidth-limited channel.

---

<sup>1</sup> In the Proceedings of the IS&T/SPIE Storage and Retrieval for Media Databases 2000, San Jose, Jan 2000.

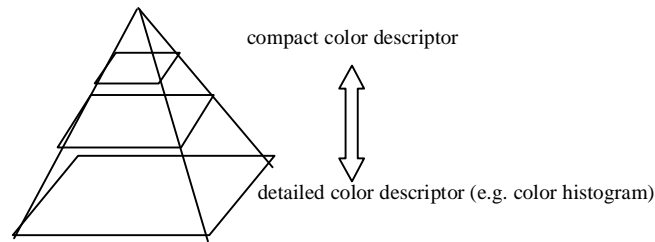


Figure 1: Color descriptor hierarchy

MPEG-7<sup>4</sup> is an ongoing standardization activity for multimedia content descriptions. The goal of the standard is to come up with a representation for content description to facilitate interoperability between content providers and consumers. There are many applications that can benefit from the standard. Two main categories of these applications are *push* and *pull* applications. Push applications involve filtering content. One example is including filtering capabilities with the TV at the consumer's end. This filtering can be done using the consumer's profile and the MPEG-7 descriptions. Pull applications involve searching for content from large volumes of digital content; one example is searching through archives of content in a broadcast library or over the Internet.

This paper presents a compact descriptor based on color histograms. This descriptor is obtained by applying the Haar wavelet transformation to a color histogram and binary quantizing the resulting coefficients of the transform. The resulting descriptor is 63 bits long and is therefore very compact. The simplicity of the Haar transform allows easy computation of the proposed descriptor. In spite of its compactness, the descriptor performs well for image, video search and browsing tasks. Objective measures developed by the MPEG-7 group to evaluate various representations show that this descriptor offers a superior performance when compared to other descriptors of comparable size. The binary nature of the descriptor allows simple matching metric based on Hamming distance and simple indexing scheme. The computationally inexpensive similarity metric associated with the descriptor makes it possible to perform very fast image search and video segment search without any shot segmentation.

The paper is organized as follows. The details of the binary Haar histogram are presented in Section 2. Section 3 carries the details of a simple indexing scheme with this descriptor. Section 4 presents the details of the performance of the descriptor for image matching based on the objective measures developed by MPEG-7. Section 5 presents the details of video segment matching. Section 6 presents the details of a browsing technique developed using this descriptor. Section 7 summarizes and concludes the paper.

## 2. BINARY HAAR HISTOGRAM

The feature extraction for the binary Haar histogram includes two stages. The first stage is obtaining the color histogram. Using a suitable color space and quantization table the color histogram is obtained. The histogram can be one, two or three-dimensional histogram. The second stage in the feature extraction is the computation of the Haar transform coefficients of the histogram. The second stage does not depend on the nature of the color space or the quantization table used in the first stage. Figure 2 shows the eight basis functions of the Haar wavelet of length eight. The Haar wavelet coefficients are obtained by taking the inner product of the basis functions with the given histogram.

The Haar wavelets are computationally very simple to implement, because the basis functions contain only values of +1 or -1. Therefore, the computation does not involve any multiplication. The Haar coefficients capture the *qualitative* aspects of the histogram. For example, the second coefficient (from the basis function 2 in Figure 2) is positive if the sum of the left half of the histogram bins is greater than the right half and negative otherwise. Similarly, the third coefficient is positive if the sum of first quarter of the histogram bins is greater than the second quarter and negative otherwise. In our descriptor, each of these coefficients is quantized to a 0 or 1, depending on whether their value is negative or positive and hence a binary representation is obtained. The motivation behind this proposal is that for search and filtering applications, exact histogram values are not necessary, but a general qualitative description of the histogram is sufficient. Also note that the first coefficient

is always 1. Since the first coefficient corresponds to sum of all probabilities in a histogram, it is always positive and therefore is quantized to 1. Therefore, this coefficient is not used in similarity matching.

Two different color spaces have been used for computing the color Histograms, the YCbCr and the HSV color space. We have used a 64-bin color histogram, but the technique is equally applicable to histograms of larger or smaller number of quantization bins. The quantized color centers were obtained based on the color probabilities of the image data set. The Haar transformation is implemented as follows. The histogram values corresponding to the 64 quantization centers are placed on a two dimensional 8-by-8 grid. The Haar transform coefficients are computed in a hierarchical fashion. In the first level, the 8-by-8 grid of histogram values is divided into two halves vertically. The sum of the histogram values in the left half is subtracted from the sum of histogram values in the right half. The resulting first Haar coefficient is quantized to 1 if it is greater than 0 and is quantized to 0 otherwise. The process is then repeated recursively, *i.e.*, in the second level, the left (right) half of the 8-by-8 grid is split horizontally into two 4-by-4 halves and the sum of histogram values in one half is subtracted from the other resulting in the second (third) Haar coefficient. The resulting second (third) Haar coefficient is then binary quantized. This is repeated recursively at the third, fourth, fifth, and sixth levels resulting in 4,8,16, and 32 coefficients respectively. Therefore, there is a total of 63 (1+2+4+8+16+32) binary Haar coefficients that form the compact descriptor. More details on various implementations of Haar transform can be found elsewhere<sup>5</sup>. The details of the quantization used for the different color spaces have been presented in an MPEG input document<sup>6</sup>.

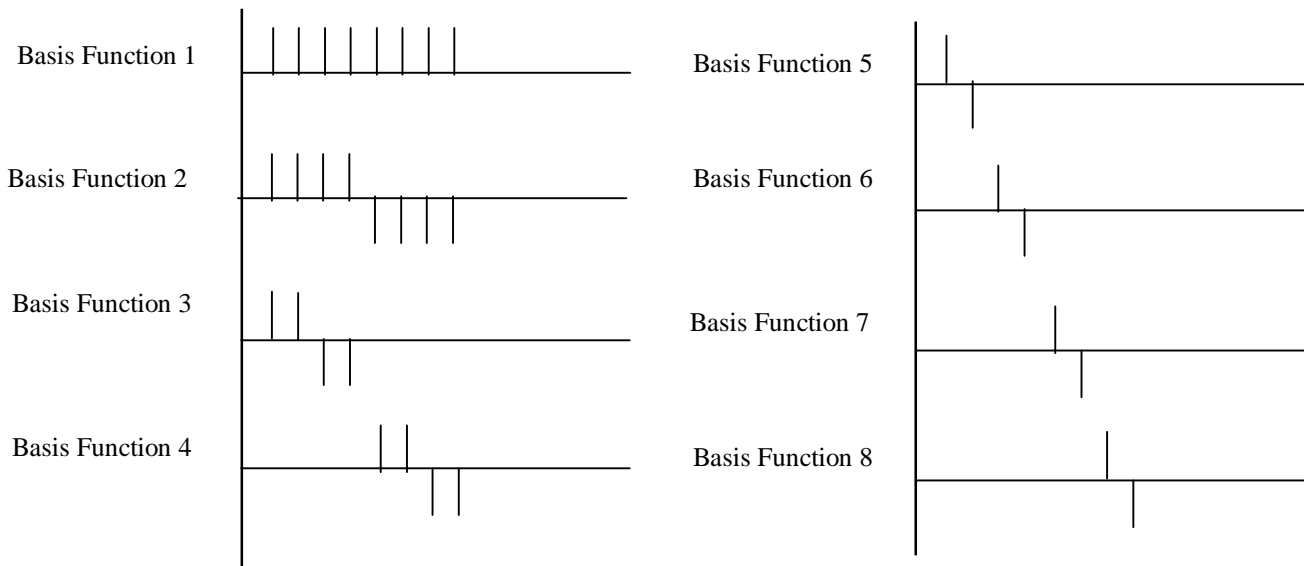


Figure 2: The basis functions for the Haar wavelet transform

### 3. INDEXING

As the size of the database grows larger, it is necessary to index the contents so that an efficient search can be performed. Conventional histograms with large bin sizes are hard to index efficiently. As the dimensionality of the feature grows beyond 20 or so, efficient indexing becomes very difficult<sup>7</sup>. The binary nature of the Haar histogram is amenable to very efficient and simple indexing schemes. Figure 3 shows an example of an indexing scheme using a binary tree. At the root of the tree, the first bit of the binary Haar histogram is used to decide whether the right or the left path is chosen. At the next level, the second bit of the binary Haar histogram is used to decide the same and so on and finally the given image is inserted at the leaves of the tree. The depth of the tree depends on the number of bits in the descriptor. With such an indexing scheme, it is possible to find very efficiently images that have the same binary Haar histogram as a query image.

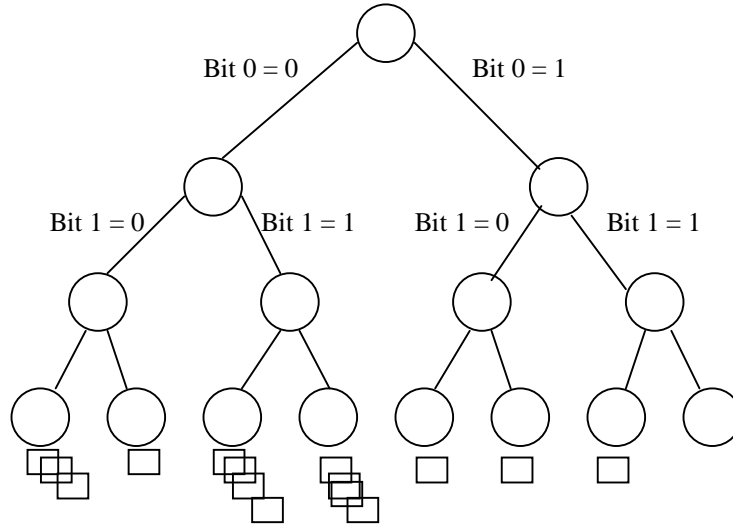


Figure 3: An indexing scheme for the binary Haar histogram descriptor

#### 4. IMAGE MATCHING

The similarity metric used for image and video segment matching using the binary Haar histograms is the Hamming distance. The Hamming distance is computed by taking the XOR of the two 63-bit binary descriptors and computing the number of '1' bits in the result. The image matching tests were performed with the content and test set chosen by MPEG-7 adhoc group on color and texture core experiments. A database of 5466 images was used. The images are chosen from many different sources and some of the images are sampled from video sequences. The description for each of these images is pre-computed and stored. When one of the database images is selected as the query, its description is compared with the description of all the database images using the Hamming distance and the results are ordered and presented.

To evaluate the performance of various descriptors, MPEG-7 developed an objective measure. To use this measure, a set of 50 query images and corresponding ground truth were obtained. The query and ground truth sets were chosen manually. For each query a set of ground truth images that are most relevant to the query were identified. The ground truth images are not ordered in any way, but all ground truth images were deemed to be equally relevant to the query image. A good descriptor is expected to retrieve all the ground truth images for a given query image.

The objective measure used by MPEG-7 to evaluate the retrieval performance is presented here. This measure combines the precision and recall measure to obtain a single objective value.

Let the number of ground truth images for a given query  $q$  be  $NG(q)$

- First  $K$  retrievals (the top ranked  $K$  retrievals) are examined, where  $K = \min(4 * NG(q), 2 * GTM)$  and  $GTM$  is  $\max\{NG(q)\}$  for all  $q$ 's of a data set.
- For each ground truth image  $k$  that was retrieved in the top  $K$  retrieval, a rank value  $Rank(k)$  is attached. The  $Rank(k)$  is the retrieval rank of the ground truth image. The rank of the first retrieved item is one.
- A Rank of  $(K+1)$  is assigned to each of the ground truth images, which are not in the first  $K$  retrievals.
- The average rank  $AVR(q)$  for query  $q$  is computed as follows:

$$AVR(q) = \sum_{k=1}^{NG(q)} \frac{Rank(k)}{NG(q)}$$

- The modified retrieval rank is computed as follows:

$$MRR(q) = AVR(q) - 0.5 - \frac{NG(q)}{2}$$

The modified retrieval rank has the property that for perfect retrieval, i.e., when the  $NG(q)$  ground truth images are the top retrievals, the  $MRR$  value is 0.

- The normalized modified retrieval rank is computed as follows:

$$NMRR(q) = \frac{MRR(q)}{K + 0.5 - 0.5 * NG(q)}$$

Note that *the*  $NMRR(q)$  will always be in the range of  $[0.0, 1.0]$ .

- Compute average of  $NMRR$  over all queries

$$ANMRR = \frac{1}{Q} \sum_{q=1}^Q NMRR(q)$$

The average normalized modified retrieval rank ANMRR is used as the objective measure to evaluate the retrieval performances. The ANMRR is always in the range of 0 to 1.0 and smaller the value of this measure, the better the quality of retrieval. Using the above measure, the retrieval performance of the binary Haar histogram is compared with the conventional 64-bit binary histogram. The conventional 64-bit binary histogram is obtained by computing the 64-bin histogram and binary quantizing each bin value. The same quantization centers were used in both cases. Table 1 shows the comparison of performances in two color spaces.

	YCbCr	HSV
Binary Haar Histogram	0.207	0.241
Conventional Binary Histogram	0.423	0.487

The objective measure clearly shows the vastly improved performance of the binary Haar histogram compared with the conventional binary histogram of the same size. Figure 5 shows the result of the search using the binary Haar transform for one of the queries.

## 5. VIDEO SEGMENT MATCHING

Typically video segment search is performed after detecting the shots in the videos. However, shot detection by itself is fairly difficult and often results in incorrect shot boundaries. Here the video segment matching is performed without any shot detection. Given a query video segment, it has to be compared with all video segments in the database with all possible starting points since there is no shot detection. This considerably increases the number of similarity matches to be performed. However, because of the compactness of the descriptor and the simplicity of the matching function this can be performed very efficiently.

To perform video segment search without shot detection, we extracted a key-frame every half a second from each video sequence. When the user selects a query video segment, again, key-frames are selected every half a second from the query segment. The query key-frames are compared with each video in the database in a time-overlapped fashion as shown in Figure 4.

The query to the database is a video segment with a given starting time and length. Since we selected two key-frames per second, the starting time and the length of the segment can be expressed with a resolution of  $\frac{1}{2}$  second. Given the starting time and the length of the segment, the corresponding key-frames are identified. If the length of the segment is  $L$  seconds, then  $2L$  key-frames are chosen.

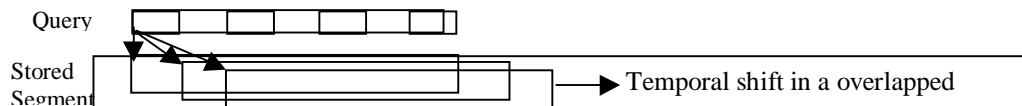


Figure 4: Video segment matching without shot detection

The comparison of the query segment with a video sequence  $V$  is performed as follows. Let there be  $K$  key-frames obtained from a video sequence, numbered from 0 to  $K-1$  in temporal order. Typically the value of  $K$  is much larger than  $L$ . First, the  $2L$  query key-frames are compared with the first  $2L$  key-frames ( 0 to  $2L-1$ ) of  $V$ , by comparing the corresponding key-frames using the Hamming distance and computing the cumulative sums of the Hamming distances over the  $2L$  key-frames to obtain the total measure of dissimilarity between the two segments. Then the  $2L$  query key-frames are compared with the



Figure 5: A sample result of image matching using binary Haar histogram descriptor

key-frames numbered 1 to  $2L$  of  $V$ . This is repeated until the comparison with key-frames numbered  $K-2L$  to  $K-1$  is finished. This is repeated for all the video sequences. Using the computed dissimilarity values, the video segments are ordered and displayed to the user. Overlapping video segments in the results are removed and then displayed to the user.

The content set for video segment search contains 14 different video sequences. The lengths of the individual sequences vary from 5 minutes to 45 minutes. Together the 14 different video sequences make up about 4 hours of video. For each video sequence, 2 frames per second are selected as key-frames. A total of close to 30,000 key-frames are obtained from different video sequences. Queries are chosen from the video sequences and are typically of length 2 to 10 seconds. A 10-second query (20 key-frames) require about  $20 \times 30000 = 600,000$  similarity comparisons. We found that the search over the 4-hour

database can be performed in about 5-6 seconds, because of the simplicity of the similarity metric and the compactness of the descriptor.

Figure 6 shows the results of a video segment matching. We found that it is possible to make a higher level semantic segmentation of the video sequences using this descriptor. For example, Figure 6 shows a query where a new anchor makes an initial announcement of news segment before the actual news segment begins. We found that by inputting one such segment as a query we are able to detect all such other segments in the news program. This can be used to parse the news program into anchor shots and the actual news segments. In general, most news programs use the same camera position and background while showing anchor person shots, hence a color based search can easily retrieve such similar shots. Similarly the graphics that appear before the news (and other) programs are always the same. This can be used to detect the starting of a given program and the identification of a TV program.



Figure 6: Results of video segment search. A 9-second query of the news anchor and the resulting

## 6. BROWSING

Most of the content-based search systems require the user to perform search by example, by choosing one of the existing images or video segments in the database as a query or by requiring the user to compose a query. Often users cannot present to the system good examples of what they are looking for without effectively browsing through the database. Hence an efficient browsing tool is essential to introduce the user to the contents of the database. After identifying an image or a video segment of interest through browsing, the user can perform content-based search to find similar entities from the database.

For databases with large numbers of images, it is not feasible to browse linearly through the images in the database. A desirable characteristic of a browsing system is to let the user navigate through the database in a structured manner. Browsing tools for video material<sup>8,9</sup> are based on detecting shots and identifying key-frames corresponding to shots. The selected key frames are then grouped in temporal order or clustered based on content similarity to aid browsing. Browsing tools for large image databases have been presented using clustering techniques. An active browsing tool<sup>10</sup> that uses hierarchical clustering and relevance feedback has been presented recently. The use of hierarchical clustering technique for fast image search and non-linear browsing<sup>11</sup> has also been shown.

The drawbacks of the browsing techniques that use clustering techniques are : (1) they require computationally expensive clustering algorithms, (2) addition or deletion of content from the database may change the clusters considerably requiring re-clustering periodically. The browsing technique presented here does not require any clustering and is automatically obtained from the binary Haar histogram descriptor. The proposed technique is not affected by the addition or removal of content from the database.

A balanced  $2^k$ -ary tree is constructed for browsing. At the first level, a set of  $k$  bits out of 63 bits in the binary Haar histogram descriptor are chosen. Each image can have one of  $2^k$  values for these  $k$  bits. Depending on the values of these  $k$  bits, the entire database is divided in  $2^k$  groups. Each of these groups is represented by a child node from the root of the tree. Each of these groups is further divided into  $2^k$  groups, by considering another set of  $k$  bits. This process is repeated to build the entire tree. A representative images is chosen at each node in the tree. In the first level of browsing at the root of the tree, the  $2^k$  representative images corresponding to the first level of the tree are displayed. At the subsequent levels, the user traverses through the tree by selecting the representative image that is most similar to what is being looked for. The selection of the  $k$  bits that are used to partition the database into groups can be selected in two ways: (1) selecting the most significant bits first followed by the least significant bits or (2) identifying the entropy of each of the 63 bits for a given database and selecting the bits in the decreasing order of the entropy values. The reasoning behind the latter selection is that the bits with the largest entropy contain the most information and therefore are used to partition the database initially.

## 7. SUMMARY AND CONCLUSIONS

We have presented a compact color descriptor based on color histograms. The compact descriptor is obtained by computing the Haar wavelet transformation of the color histograms followed by binary quantizing the resulting coefficients. This descriptor captures the qualitative aspects of the color content of the images as opposed to the quantitative aspects captured by the conventional histogram. The resulting descriptor is very compact in size and due to the binary nature of the descriptor the associated similarity metric is very simple. Such compact descriptors complement other detailed color descriptors like histograms and are extremely useful in bandwidth constrained applications that require streaming of the content descriptors along with the media content. The efficacy of the descriptor for image matching, video segment matching without shot detection and browsing applications have been shown. Experiments performed on four hours of video show that this compact descriptor performs sufficiently well in identifying similar video segments that can be used for higher level semantic segmentation of video. Objective comparisons on image matching performed by the MPEG-7 group show that the proposed compact descriptor perform much better than other descriptors of comparable sizes.

## 8. REFERENCES

1. W. Niblack, et.al., "The QBIC Project: querying images by content using color, texture, and shape", In *Storage and Retrieval for Image and Video Databases I*, Vol. 1908, SPIE Proceedings, Feb. 1993.
2. J. R. Smith and S-F. Chang, " A Fully Automated Content-based Image Query System", *Proc. of ACM Intl. Multimedia Conference*, Nov. 1996.

3. N. Dimitrova, H. Elenbaas and T. McGee, "PNRS - Personal News Retrieval System", In *SPIE Conference on Multimedia Storage and Archiving Systems, SPIE Proceedings*, Sep. 1999.
4. M. Abdel-Mottaleb, et.al, "MPEG-7: A Content Description Standard Beyond Compression", In *IEEE Mid-west Symposium on Circuits and Systems*, Aug. 1999.
5. E.J. Stollnitz, T.D. DeRose, and D.H. Salesin, "*Wavelets for Computer Graphics - Theory and Applications*", Morgan Kaufmann Publishers, 1996.
6. S. Krishnamachari and M. Abdel-Mottaleb, "Results of the core experiment on compact color descriptors", ISO/JTC1/SC29/WG11/M5192 Melbourne, Oct. 1999.
7. Y. Gong, "*Intelligent Image Databases: Towards Advanced Image Retrieval*", Kluwer Academic Publishers, 1998.
8. N. Dimitrova, T. McGee, H. Elenbaas, and J. Martino, "Video Content Management in Consumer Devices", *IEEE Transactions on Knowledge and Data Engineering*, Nov. 1998.
9. H.J. Zhang, Y.L. Chien, and S.W. Smoliar, "Video Parsing and Browsing Using Compressed Data," *Proc. Multimedia Tools and Applications*, Vol. 1, No. 1, pp. 89-111, 1995.
10. J-Y. Chen, C.A. Bouman and J. Dalton, "Active Browsing using Similarity Pyramids", In *Storage and Retrieval for Image and Video Databases VII*, Vol. 3656, SPIE Proceedings, Jan 1998.
11. S. Krishnamachari and M. Abdel-Mottaleb, "Image Browsing using Hierarchical Clustering", In *Proc. of the IEEE Symposium on Computers and Communications, ISCC 99*, Egypt, July 1999.