Background Removal in Image indexing and Retrieval

Yi Lu and Hong Guo
Department of Electrical and Computer Engineering
The University of Michigan-Dearborn
Dearborn Michigan 48128-1491, U.S.A.
Voice: 313-593-5028, Fax: 313-593-9967
yilu@umich.edu

Abstract

This paper presents our research in image content based indexing and retrieval, a key technology in digital image libraries. In most of the existing image content-based techniques, image features used for indexing and retrieval are global, features are computed over the entire image. The major problem with the global image feature based retrieval methods is that background features can be easily mistakenly taken as object features. When a user attempts to retrieve images using color features, he/she usually means the color feature of an object or objects of interests contained in the image. The approach we describe in this paper utilizes color clusters for image background analysis. Once the background regions are identified, they are removed from the image indexing procedure; therefore, no longer interfering with the meaningful image content during the retrieval process. The algorithm consists of three major steps of computation, fuzzy clustering, color image segmentation, and background analysis.

1. Introduction

Digital imagery is a convenient medium for describing and storing spatial, temporal, spectral and physical components of information contained in a variety of domains. Large databases contain thousands of digital images that can occupy gigabytes of space, and hence it is difficult for a user to find useful information using simple browsing techniques. Therefore, efficient and automatic algorithms are required for indexing and retrieving images from databases. Conventionally images are indexed using text information, such as keywords, date, artist, etc. [Gro88, Nag85, FJL97]. However, the inadequacy of textual descriptions is obvious because the complexity of the information imbedded in images (number and types of objects, their attributes and spatial relationships) can not be synthesized in a few key words. Until recently with the fast growing technology in computer vision, image content based indexing and retrieval are the new areas of research and development [Jai93, PPS94, FSN95, Gup6, FDW96]. There is a growing demand for systems using pictorial information for both commercial and cultural applications.

Research in content-based image retrieval can be characterized in the following categories: query-by-visual sample, pictorial queries, and/or linguistic queries. In a query by visual sample, a user must submit a sample image to the system, and then the system searches its database to find the images most "similar" to the sample image. The similarity measure is usually taken from image features such as color, shape and texture. The IBM QBIC system [FSN95] and the Virage image search engine both use this type of query[Gup96]. Pictorial queries are sketch-based queries, and these systems usually provide sketch tools to allow the user to draw image features, such as shape, size, color, and texture that occur in the images to be retrieved. The system uses the sketch as a query and search for images that contain the sketch. VisualSEEK, a content-based image/video retrieval system at Columbia University [SmC96], uses pictorial queries extensively. Belongie et al[BCG98] used blobworld to represent the composition of image content. The blobworld descriptors uses 2-D ellipses, or "blobs", each of which possesses a number of attributes. Linguistic queries that directly refer to image contents are popular among many users. These queries clearly identify salient visual properties contained in the desired images using the languages are understandable by most population. These image features can be low level, such as “red,” or higher level such as “horizon” or “horse.” The Digital Library Initiative at the University of California at Berkeley has made a significant effort in this direction[Wil96].

In most of the existing image content-based techniques, image features used for indexing and retrieval are global features, features are computed over the entire image. For example, retrieving similar images in color, the most of the existing techniques use a color histogram generated from the entire image[HSE95, JAV96]. The major problem with the global color histogram based retrieval methods is that they do not possess spatial relationship between object regions and it does not differentiate the color between the background and foreground. When a user attempts to retrieve images using color features, he/she usually means the color feature of an object or objects of interests contained in the image. For example, if a design engineer wants to retrieve the images of automobiles using color feature, he would likely submit queries such as “red cars” or “blue seat covers,” etc. Therefore in many applications, it makes more sense to index and retrieve images based on the color information of objects. If a retrieval system only uses color as the feature for image retrieval, a user may get many red cars with black background for a given query “black car.”

The differentiation of background from foreground is an ill-posed problem in image analysis. For motion pictures, the background features in the consecutive images do not change much, therefore we can use the difference image of two consecutive images in the time domain to differentiate background from foreground. In still images, image background is not well defined. Generally speaking, the background of an image includes all the features in the image that do not belong to the objects of interests. In many images, background can be easily recognized by a human being but difficult for computer, because the human beings have excellent object recognition and analysis abilities. In many images image background features are even ambiguous to human observers. Our approach to this problem is to eliminate regions that are certain to be backgrounds. For those images that are ambiguous to human perception, we will not interfere. We do not intend to use object recognition in image retrieval, which is a very complicated classic computer vision problem. Instead, we are investigating a computationally efficiency and yet effective background removal algorithm that can facilitate image retrieval.

The approach we describe in this paper utilizes color clusters for image background analysis. Once the background regions are identified, they are removed from the image indexing procedure; therefore, they are no longer interfering with the meaningful image content during the retrieval process.

We have implemented the algorithm at our web site, and tested the system on images from three different collections of images, car images, Earth and space images, and crashed car images. We have compared the algorithm with a global color histogram based approach and the results show that the algorithm with background removal is very effective in retrieving images containing a few major objects. Due to the lack of color printing in this publication, we have reduced number of image examples and converted all color images to gray scales. A color version of the paper can be found at website:[Guo: put the address here].

2. Background region analysis

Figure 1 gives an over view of our system that performs background region analysis. The system has three major computational components: fuzzy clustering algorithm, color image segmentation, and background region analysis. The fuzzy clustering algorithm attempts to find similar colors in the L*u*v* space using fuzzy logic. The algorithm considers a cluster of similar colors as a fuzzy set, and represents the likeliness of a color pixel belonging to a fuzzy set by a fuzzy membership

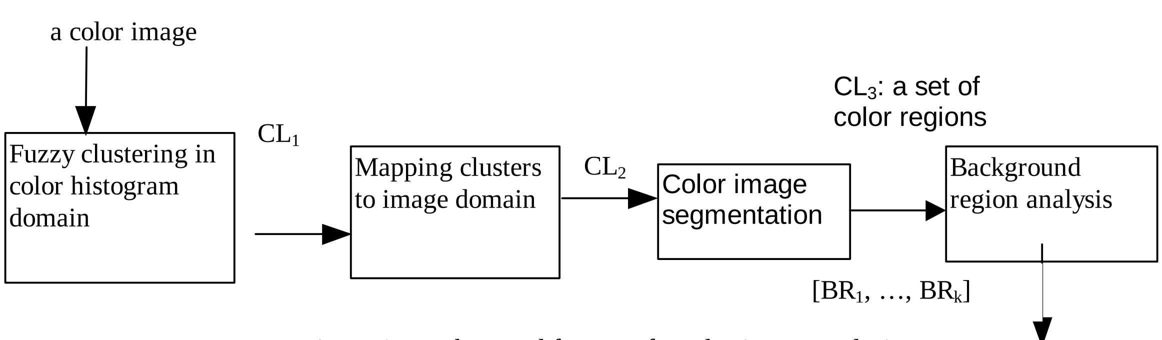


Figure 1. Background feature of a color image analysis

function. The fuzzy clustering algorithm yields a decomposition of the 3D-color histogram; namely a number of non-overlapping heaps in the 3D-color histogram, CL_1. The labeling of the clusters in CL_1 results in a multithresholded image. The details of this algorithm can be found in [ChL98]. In this paper, we will only discuss the color image segmentation and the background analysis algorithm, and how the system can be used in image content based indexing and retrieval.

2.1 Color region segmentation

The fuzzy clustering algorithm mentioned above generates a set of clusters, CL_1, each of the clusters in CL_1 represents a sub-spectrum of colors in the L*u*v* space. The mapping of clusters in CL_1 to the image domain takes two steps. First we compute the central color of each cluster in CL_1 using the formula below.

Assume M clusters in CL_1 and each image pixel p_i belongs to one of the clusters. C(p_i)=K denotes p_i belongs to the *Kth* cluster, and its L*u*v* values are [l_i, u_i, v_i]. The central color of cluster K is represented by [$l^{(K)}, u^{(K)}, v^{(K)}$], which is calculated as follows:

$$l^{(K)} = \frac{\sum_{\substack{i \\ C(p_i)=K}} l_i}{\#\text{of pixels in cluster K}}$$

$$u^{(K)} = \frac{\sum_{\substack{i \\ C(p_i)=K}} u_i}{\#\text{of pixels in cluster K}}$$

$$v^{(K)} = \frac{\sum_{\substack{i \\ C(p_i)=K}} v_i}{\#\text{of pixels in cluster K}}$$

Each pixel in the image domain is assigned the representative color value of its cluster. We obtain the second set of clusters, CL_2, by computing the connected pixels of the same cluster color. In general, one cluster in CL_1 may generate more than one clusters in CL_2, therefore, CL_2 is usually much larger than CL_1. For example, in one of our experiments, the algorithm generated, from a CL_1 with |CL1| = 64, the CL_2 that contains more than five hundred clusters. The clusters contained in CL_2 have the following properties: Pixels in the same cluster are spatially connected and are assigned of the same color: the representative color of the cluster. Therefore, we sometimes refer the clusters in CL_2 as color regions. The regions in CL_2 are usually small and a meaningful image region contains several clusters in CL_2.

The next step in the image segmentation is an agglomerative process. It attempts to merge clusters in CL_2 based on

- the color distances among neighboring clusters in the spatial domain,
- cluster sizes and
- the maximum number of clusters in CL_3.

During the agglomerative process, the distance between two clusters is defined as the color distance between the representative colors of the two clusters, and the color distance functions we use are defined in the L*u*v* space. For two dark colors, i.e. the colors whose L*, u* and v* components all have very small values, then color distance is dependent on the L component only, i.e.,

$$dist([L_1^*, u_1^*, v_1^*],[L_2^*, u_2^*, v_2^*]) = \left|L_1^* - L_2^*\right|.$$

For other colors, we use the Euclidean distance. It is important to use a different distance function for dark colors. As we found out empirically that human perception does not match the Euclidean distance to differentiate dark colors. For example in Figure 2, region A and region B have very close colors according to human perception and, therefore, they should be regarded as belonging to the same region. However, we found that the Euclidean distance between the two regions is large. For example, one typical L*u*v* value of a pixel in region A is (39.45,11.96,-27.32), and one typical L*u*v* value of a pixel in region B is (30.87,27.22, -4.78), and hhe Euclidean distance of the two color pixels is *dist* = 28.54, which is quite large. However, the dark color distance value is *dist* = 8.58, which is more consistent with human perception. If we used the Euclidean distance function in the image segmentation, the background region in this image is very likely to be divided into several smaller regions. This example suggests that for dark colors, the saturation components are insignificant.

Figure 2. Illustration of dark color distance function. Region A and B look similar from human perception. While their L*u*v* distance is large.

We have explored the following four different methods in merging clusters. In the first three methods, a common parameter, *max_cls* is used to control the maximum number of clusters in CL3. Method 1 considers the clusters that are adjacent and have similar colors as the first candidates for merging. It uses a control parameter, *cl_diff_th* to denote color difference threshold. It first attempts to merge the neighboring clusters whose color distances are below cl_diff_th. The order of merging is irrelevant. If the number of clusters at the end of the merging process is greater than *max_cls*, then the algorithm selects the smallest cluster and merges the cluster with one of its neighbors to which it has the smallest color distance. This merging process repeats until the number of clusters in CL3 is no more than *max_cls*. The second method considers the size of clusters as the only merging criterion. It selects the smallest cluster and merges the cluster with one of its neighbors to which it has the smallest color distance. The process is repeated until the number of clusters in CL3 is no more than *max_cls*. The third method considers the color distance as the most important criterion in spatial clustering. However, the computation in finding the minimum color distance between two adjacent clusters is quite time consuming if a large number of clusters exist. To alleviate the computational burden, we first repeatedly merges the smallest clusters with their neighbors in the closest color distance until the total number of clusters is reasonable, e.g. 100. In the second pass, we select the two adjacent clusters whose color distance is the smallest within the entire image and merges the two clusters. This process repeats until the top max_cls clusters in size contain a large percentage of the image pixels (e.g. 99%). Since the largest max_cl clusters already cover the major percent of the entire image pixels, the small clusters below the top max_cl do not affect much the final segmentation result. Therefore at the third pass, the algorithm repeatedly merges the smallest cluster with its closest neighbor in color distance until the total number of clusters in CL_3 is no more than max_cls. The last method we explored used two different thresholds for color distance functions. For clusters that have smaller size, they are merged if their color distance is less than the larger threshold, and for larger clusters, they are merged if their color distance is less than the smaller threshold. The threshold for small clusters is determined by the image size. In all these methods, the center of the new region is calculated after every merge, the size of the region has changed, and so do the neighbors. The representative color of a new region merged from C_1 and C_2 is calculated using the following formula:

$$L_N^* = \frac{L_1^* * N_1 + L_2^* * N_2}{N_1 + N_2}$$

$$u_N^* = \frac{u_1^* * N_1 + u_2^* * N_2}{N_1 + N_2}$$

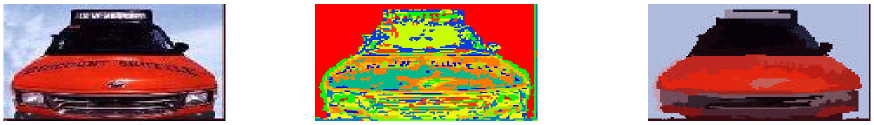
$$v_N^* = \frac{v_1^* * N_1 + v_2^* * N_2}{N_1 + N_2}$$

where (L_1^*, u_1^*, v_1^*) and (L_2^*, u_2^*, v_2^*) are the representative colors of C_1 and C_2 respectively. Note the order of merging may affect the clustering result in the image domain. The results from the spatial clustering is a set of color regions, CL_3, in which all the pixels in the same region in CL_3 are spatially connected and have the same representative color.

From computational point of view, Method 1 and 2 are more efficient than Method 3. Method 4 is the more efficient in terms of computation than method 3 and more effective than Method 1 and 2. However, in many cases, Method 3 generates better results than the others do. In the trade off of computational time and segmentation results, we chose to use method 4 in our background analysis algorithm. Figure 4 shows an example of image segmentation results generated by Method 4. Figure 4 (a) shows the original image, (b) shows the image of clusters generated by mapping the clusters generated by the fuzzy clustering algorithm directly to the image domain, and (c) shows the output from the segmentation algorithm. For the purpose of demonstration, the image shown in (b) is drawn in pseudo colors. The colors shown in (c), the resulting image, are true representative colors of color regions. It can be seen, the significant color features are well captured after color image segmentation.

2.2 Image background analysis

The background analysis algorithm is developed based on the following hypotheses:



(a) (b) (c)

Figure 3. Example 1 of image segmentation. (a)The original image. (b) The image after the color clustering. There are 12 color clusters and 598 spatial clusters. (c) The segmentation result.

- A background region must be moderate in size and must be connected to the image borders. The main purpose of this algorithm is to identify big color regions that are not belonging to objects of interests, since only large background regions can hamper the retrieving results.
- Regions occupy image corners are more likely to be background.
- For a given image collection, we may have specified background colors. However, since a background color can also appear in objects of interests in the images of the same collection, and not all the images have the same background colors, the prior knowledge of background colors should be used cautiously.

Assume there are M color regions in the image, denoted by R_1 , R_2 , … $R_{M.}$ Every region is described by { C_i, $x_{i1,}$ $y_{i1,}$ x_{i2} ,y_{i2}}, where C_i is the representative color of the ith region and ($x_{i1,}$ y_{i1}), (x_{i2} ,y_{i2}) are the coordinates of the top-left and bottom-right corners of the bounding box of the region. We first divide the image plane into nine blocks shown in Figure 4.

0	1	2
3	4	5
6	7	8

Figure 4. An image plane is divided into nine blocks for background analysis.

The sizes of the nine blocks are determined by the image size. In general we want the center block to be large enough to contain the image contents. Based on our experience, the following sizes of the nine blocks are feasible. Let the width and height of an image be W and H respectively. The center block 4 is 0.7W*0.7H, the four corner blocks, 0,2, 6 and 8, have size of 0.15H*0.15W, the four lateral blocks, 1 and 7, have size of 0.7W*0.15H, 3 and 5 of 0.15Hx0.7W.

For each region $R_{j,}$ we calculate its percentage r_{ji} over each image block i , (i=0, 1,….8). From the percentage, we can obtain the number of corners and lateral regions one cluster occupies. Let N_c and N_l be the number of corners and the number of lateral regions a cluster occupies respectively. We set these two variables as follows:

N_c = the number of corner regions whose occupancy ratio is greater than a given threshold q_a .

N_l = the number of lateral regions whose occupancy ratio is greater than a given threshold.

The algorithm can be summarized into the following rules.

Rule1. For a region $R_{j,}$ if its representative color C_j belongs to the specified background color and the region occupies more than two corners, i.e. N_c>2, then R_j is marked as a background region.

Rule2. For a region $R_{j,}$ if its representative color C_j belongs to the specified background color and the region occupies more than one corners and one lateral blocks, i.e. (N_c>0 and N_l>0), then R_j is marked as a background region.

These two rules apply to an image collection with specified background colors. In certain applications this specification is possible. For examples, images taken indoors often have specific background colors; in our Earth and Space Science image collection, we found that "black" and "white" often appear as background colors.

For those image collections have no specified background colors, we use the following rules:

Rule3. For an image region $R_{j,}$ if it occupies more than two corner blocks, i.e. N_c>2,, and its percentage over the center block is small, then set R_j as a background region.

Rule4. For an image region $R_{j,}$ if it occupies one or more corner and lateral blocks, and its percentage over the center block is small, i.e. $r_{j4} < q_b$, where q_b is a threshold, then set R_j as a background region.

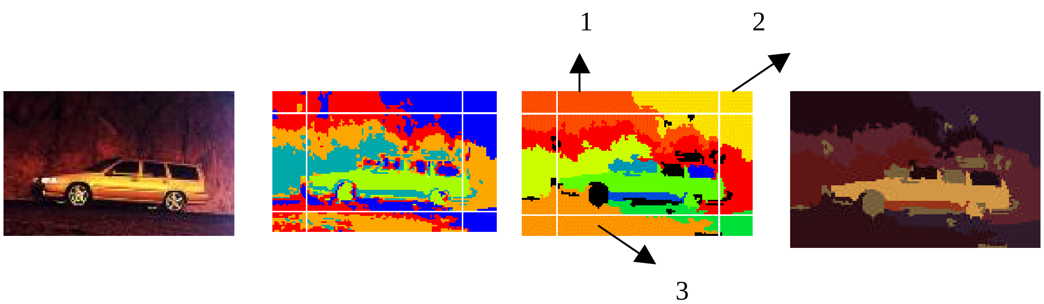
Rule5. For an image region R_j, if it occupies at least one corner or one lateral regions, i.e., $N_c>0$, or $N_l>0$, and most of R_j's pixels are in the lateral and corner blocks, then set R_j as a background region.

In many images, background regions occupy the corner and lateral blocks, and therefore, the corner and lateral regions are emphasized in the background region analysis. Additionally only large clusters can be recognized as background, which is controlled by q_a in the computation of N_c and N_l. For example, if we set q_a to 0.85, then a image region is considered as occupying a corner only if one of r_{j0}, r_{j2}, r_{j6}, r_{j8} is above 85%. In Rule 3 and 4, in addition to the occupancy measure of corners and laterals, we also check occupation percentage of the image region inside the central block. If this percentage is small, it implies that the center block contains some other larger regions and the current cluster may be regarded as background. Without this additional condition, these rules can be detrimental to the images that contain meaningful regions that occupy the central block as well as the corners and/or lateral blocks. For example, Figure 5 shows an example. The original image is a color photo of the Pacific Ocean with blue color. The large region connecting the corners and a good portion of the central block is the Pacific Ocean, which is the important content to the image. As the result of our background analysis algorithm, this image is considered to have no background regions.

Figure 6 shows an example of the background analysis algorithm. Figure 6(b) shows the image of clusters generated by the fuzzy clustering algorithms in the color space. Figure 6(c) shows the color regions after the segmentation algorithm. For the purpose of illustration we used pseudo colors to display different regions. The white grids in (b) and (c) divide the image into nine blocks. The labels in (c) indicate the background regions found by the background analysis algorithm. Figure 6 (d) shows the representative colors of regions generated by the segmentation algorithm. After the background analysis, we identified in this image three background regions, which are labeled as 1, 2 and 3 in (c). In this example, although cluster 1,2 and 3 are not connected with each other in the spatial domain, they are all considered as background because of the same representative background color, "black."

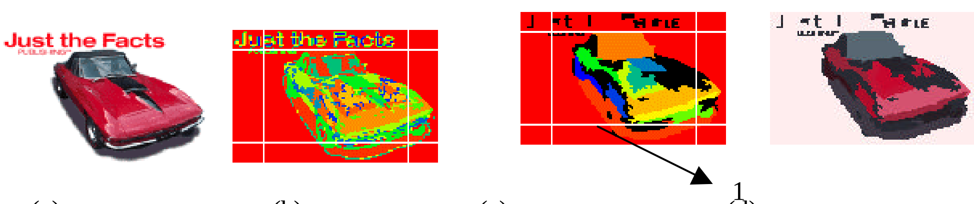
(a) original image (b) Resulting image from the Clustering algorithm

Figure 5. An image has no background regions.



(a) (b) (c) (d)

Figure 6. Example of background analysis. (a) The original image. (b) The image after of clustering in pseudo colors. (c) The color regions after segmentation in pseudo colors and 3 identified background regions. (d) The segmentation result illustrated in the representative colors.



(a) (b) (c) (d)

Figure 6. Example of background analysis. (a) Original image. (b) The image of color clusters in pseudo colors. (c) The color regions after segmentation in pseudo colors and 1 identified background region. (d) Segmentation results shown in representative colors.

[Guo: choose one of the two figures based on the quality of the converted gray scale images.]

Figure 6 shows another example of image segmentation. In this example, the algorithm found one big background region(see (c)) which has representative color "white"(see (d)).

3. Image indexing and retrieval using background analysis

We have implemented the background analysis algorithm in IMAGE-SEEKER, a content-based image retrieval system running on the web site http://frey.umd.umich.edu:8080/html/. IMAGE-SEEKER supports fifteen different colors that are commonly used in image queries. To implement these queries, we generate the color histogram using fifteen dominant colors for each image and index the image with the color histogram. During the image retrieval stage, the system takes a color query and looks for images whose dominant colors match the color query. The background analysis can be used before the color histogram generation. If the algorithm identified any background regions, we will remove these regions from the image, and then compute the color histogram from the remaining regions in the image.

Currently, our image library consists of five different image collections: images of cars, interior and exterior images of a Honda vehicle model, crashed cars, face images, and images of Earth and Space. We have tested the background analysis algorithm on all these collections using different color queries. We found the algorithm is particular effective on images of cars and crashed cars, and images of Earth and Space Science. Due to the limit of space and the absence of color printing, we are not able to show the image retrieval results in the paper. Readers may want to referred to the extended version of this paper that contains a number of examples image retrieval after background removal.

4. Conclusion

In this paper, we presented an algorithm for background region analysis in color imagery and the experiment results on the algorithm implemented in a content-based image retrieval library at a Web site. We showed that the algorithm was effectively used in content-based image retrieval on a number of different image collections. The background removal algorithm can also be used in combination with image features other than colors. For examples, we can utilize the identified background regions to index image background, differentiate in-door from out-door images, achieve more accurate retrieval using shape or texture after removing the identified background regions.

5. References

[BCG98] Serge Belongie, Chad Carson, Hayit Greenspan, and Jitendra Malik, "Color- and Texture-Based Image Segmentation Using EM and Its Application to Content-Based Image Retrieval," ICCV, 1998.

[ChL98] Tie Qi Chen and Yi Lu, "Color Image Segmentation in Color and Spatial Domains" submitted to IEEE Trans. PAMI, 1998

[FDW96] Usama M. Fayyad, S. G. Djorgovski, and Nicholas Weir, "From Digitized Images to Online Catalogs," AI magazine, pp. 51 - 66, 1996.

[FJL97] C. O. Frost, R. Johnson, S. Lee, and Y. Lu, "Research on Human-centered Design and Evaluation of Multimode Image Retrieval," Fourth DELOS Workshop San Miniato, August 28-30 1997

[FSN95] M. Flickner, H. Sawhney, W. Niblack and etc, "Query by Image and Video Content," IEEE Computer, pp. 22 - 32, September 1995.

[Gro88]W. I. Grosky, An architecture for integrating images and text in a database environment, in: *Database machines*, A. K. Sood and A. H. Qureshi, (edt.), NATO ASI Series, vol. F24, 1988

[Gup96] Amarnath Gupta, "The Virage Image Search Engine: An open framework for image management," SPIE conference on "Storage and Retrieval for Still Image and Video Databases IV," Feb. 1, 1996.

[HSE95] J. Hafner, H. S. Sawhney, W. Equitz, et al., Efficient color histogram indexing for quadratic form distance functions, IEEE trans. on PAMI, Vol. 17, No. 7, pp. 729-736, July 1995

[Jai93] R. Jain, "NSF Workshop on Visual Information Management Systems," SIGmod Record, vol. 22, No. 3, Dec. 1993, pp. 57 - 75

[JAV96] A. K. Jain and Aditya Vailaya, Image retrieval using color and shape, *Pattern Recognition*, vol. 29, no. 8, pp. 1233-1244, 1996

[Nag85] G. Nagy, Image database, *Image and Vision Computing*, vol. 3, no. 3, pp. 111-117, 1985

[PPS94] A. Pentland, R. W. Picard, and S. Sclaroff, "Photobook; Tools for Content-Based Manipulation of Image Databases," Proc. Storage and Retrieval for Image and Video Databases II, Vol. 2, 185, SPIE 1994, pp. 34 - 47.

[SmC94] J. R. Smith and Shih-Fu Chang, Quad-tree segmentation for texture-based image query, Proceedings of 2nd annual ACM multimedia conference, San Francisco, Ca, Oct. 1994

[Wil96] Robert Wilensky, "Toward Work-Centered Digital Information Services," IEEE Computer, pp. 37-45, May 1996