Querying Images by Content Using Color, Texture and Shape

Madhuri S. Bhadange, Yoginath R. Kalshetty

Computer Science & Engineering, SVERI's College of Engineering, Pandharpur, India

Abstract: ***The existing system uses a keyword to search an image. Suppose we have to search for an image of an apple, keyword should be "apple" & images related to apple will be displayed accordingly. These images are stored in a database and are displayed according to the keywords. The objective is to extract visual content of an image automatically like color, texture and shape. In proposed system, the user input a query image and then the system extracts the image feature and measure the distance with images in the database. An initial retrieval list is generated. The user can choose the relevant image to further refine the query and this process can be iterated many times until the user find the desired images. In this, the color is retrieved using RGB color space and histogram method. The edge detection method is used for finding shape or edges of an image. The Euclidean Distance is used for similarity measure between query image and database images. Finally, images are successively retrieved for query image.***

Keywords: CBIR, histogram

1. Introduction

Content-based image retrieval (CBIR), also known as query by image content (QBIC) and content-based visual information retrieval (CBVIR) is the application of computer vision techniques to the image retrieval problem, that is, the problem of searching for digital images in large databases. Content based image retrieval is opposed to concept based approaches.[19]

"Content-based" means that the search will analyze the actual contents of the image rather than the metadata such as keywords, tags, and/or descriptions associated with the image. The term 'content' in this context might refer to colors, shapes, textures, or any other information that can be derived from the image itself. CBIR is desirable because most web based image search engines rely purely on metadata and this produces a lot of garbage in the results. Also having humans manually enter keywords for images in a large database can be inefficient, expensive and may not capture every keyword that describes the image. Thus a system that can filter images based on their content would provide better indexing and return more accurate results.

Filter

Filtering refers to accepting (passing) or rejecting certain frequency components. For example, a filter passes low frequencies is called a Lowpass Filter. The net effect produced by a lowpass filter is to blur(smooth) an image. We can accomplish a similar smoothing directly on the image itself by using spatial filters(also called spatial masks, kernels, templates, and windows).

In this, Lowpass filter is used. The lowpass filters are used for blurring and for noise reduction. Blurring is used in preprocessing tasks, such as removal of small details from an image prior to (large) object extraction, and bridging of small gaps in lines or curves.[15]

Histogram

In a gray level image the probabilities assigned to each gray level can be given by the relation below:

$$P_r(r_k)=n_k/N \quad 0< r_k <1 \ , \ k=0,1,2\ldots\ldots L-1 \qquad (1)$$

Where r_k is the k^{th} intensity value and n_k is the numbers of pixels in the image with intensity r_k .N are the row and column dimensions of the image.

Use of Histogram

Histograms are simple to calculate. It give information about the kind(Global appearance) of images & its properties. Histogram is used for image Enhancement. Histogram is used for image Compression. Histogram is used for image Segmentation. It can be used for real time processing.

2. Related Work

Edge Detection

Edge detection refers to the process of identifying and locating sharp discontinuities in an image. The discontinuities are abrupt changes in pixel intensity which characterize boundaries of objects in a scene.

Canny Edge Detection Algorithm

The Canny edge detection algorithm is known to many as the optimal edge detector. Canny's intentions were to enhance the many edge detectors already out at the time he started his work. He was very successful in achieving his goal and his ideas and methods can be found in his paper, Canny J. F.[11] followed a list of criteria to improve current methods of edge detection. The first and most obvious is low error rate. It is important that edges occurring in images should not be missed and that there be no responses to non-edges. The second criterion is that the edge points be well localized. In other words, the distance between the edge pixels as found by the detector and the actual edge is to be at a minimum. A third criterion is to have only one response to a single edge. This was implemented because the first two were not substantial enough to completely eliminate the possibility of multiple responses to an edge. Based on these criteria, the canny edge detector first smoothes the image to eliminate and noise. It then finds the image gradient to highlight regions with high spatial derivatives. The algorithm then tracks along these regions and suppresses any pixel that is not at the maximum (nonmaximum suppression).

The gradient array is now further reduced by hysteresis. Hysteresis is used to track along the remaining pixels that have not been suppressed. Hysteresis uses two thresholds and if the magnitude is below the first threshold, it is set to zero (made a non edge). If the magnitude is above the high threshold, it is made an edge. And if the magnitude is between the 2 thresholds, then it is set to zero unless there is a path from this pixel to a pixel with a gradient above T2.

Step 1:

In order to implement the canny edge detector algorithm, a series of steps must be followed. The first step is to filter out any noise in the original image before trying to locate and detect any edges. And because the Gaussian filter can be computed using a simple mask, it is used exclusively in the Canny algorithm. Once a suitable mask has been calculated, the Gaussian smoothing can be performed using standard convolution methods. A convolution mask is usually much smaller than the actual image. As a result, the mask is slid over the image, manipulating a square of pixels at a time. The larger the width of the Gaussian mask, the lower is the detector's sensitivity to noise. The localization error in the detected edges also increases slightly as the Gaussian width is increased.

Step 2:

After smoothing the image and eliminating the noise, the next step is to find the edge strength by taking the gradient of the image. The Sobel operator performs a 2-D spatial gradient measurement on an image. Then, the approximate absolute gradient magnitude (edge strength) at each point can be found.

The Sobel operator [17] uses a pair of 3x3 convolution masks, one estimating the gradient in the x-direction (columns) and the other estimating the gradient in the y-direction (rows). They are shown below:

Table 1: The Gradient in x-direction (columns)

-1	0	+1
-2	0	+2
-1	0	+1

Gx

Table 2: The Gradient in y-direction (rows)

+1	+2	+1
0	0	0
-1	-2	-1

Gy

The magnitude, or edge strength, of the gradient is then approximated using the formula:

$|G| = |Gx| + |Gy|$

Step 3:

The direction of the edge is computed using the gradient in the x and y directions. However, an error will be generated when sumX is equal to zero. So in the code there has to be a restriction set whenever this takes place. Whenever the gradient in the x direction is equal to zero, the edge direction has to be equal to 90 degrees or 0 degrees, depending on what the value of the gradient in the y-direction is equal to.

If GY has a value of zero, the edge direction will equal 0 degrees. Otherwise the edge direction will equal 90 degrees. The formula for finding the edge direction is just:

Theta = invtan (Gy / Gx)

Step 4:

Once the edge direction is known, the next step is to relate the edge direction to a direction that can be traced in an image. So if the pixels of a 5x5 image are aligned as follows:

Table 3: Edge direction for 5x5 pixels images

x	x	x	x	x
x	x	x	x	x
x	x	a	x	x
x	x	x	x	x
x	x	x	x	x

Then, it can be seen by looking at pixel "a", there are only four possible directions when describing the surrounding pixels - 0 degrees (in the horizontal direction), 45 degrees (along the positive diagonal), 90 degrees (in the vertical direction), or 135 degrees (along the negative diagonal). So now the edge orientation has to be resolved into one of these four directions depending on which direction it is closest to (e.g. if the orientation angle is found to be 3 degrees, make it zero degrees). Think of this as taking a semicircle and dividing it into 5 regions.

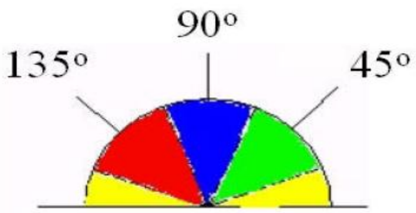


Figure 1: Edge direction falling with their ranges

Therefore, any edge direction falling within the yellow range (0 to 22.5 & 157.5 to 180 degrees) is set to 0 degrees. Any edge direction falling in the green range (22.5 to 67.5 degrees) is set to 45 degrees. Any edge direction falling in the blue range (67.5 to 112.5 degrees) is set to 90 degrees. And finally, any edge direction falling within the red range (112.5 to 157.5 degrees) is set to 135 degrees.

Step 5:

After the edge directions are known, non-maximum suppression now has to be applied. Non-maximum suppression is used to trace along the edge in the edge direction and suppress any pixel value (sets it equal to 0) that is not considered to be an edge. This will give a thin line in the output image.

Step 6:

Finally, J.Canny[12] defines hysteresis is used as a means of eliminating streaking. Streaking is the breaking up of an edge contour caused by the operator output fluctuating above and below the threshold. If a single threshold, T1 is applied to an image, and an edge has an average strength equal to T1, then due to noise, there will be instances where the edge dips below the threshold. Equally it will also extend above the threshold making an edge look like a dashed line. To avoid this, hysteresis uses 2 thresholds, a high and a low. Any pixel in the image that has a value greater than T1 is presumed to be an edge pixel, and is marked as such immediately.

Then, any pixels that are connected to this edge pixel and that have a value greater than T2 are also selected as edge pixels. If you think of following an edge, you need a gradient of T2 to start but you don't stop till you hit a gradient below T1.

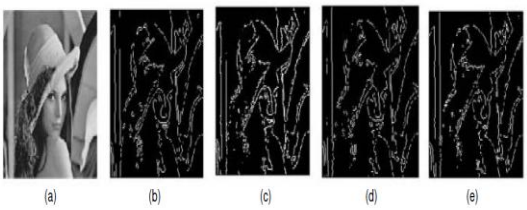


Figure 2: Comparison of Edge Detection Techniques on (a) Lena Image Original Image (b) Canny Method (c) Roberts Edges (d) LOG edges (e) Sobel

Similarity/Distance Measures

Instead of exact matching, content-based image retrieval calculates visual similarities between a query image and images in a database. Accordingly, the retrieval result is not a single image but a list of images ranked by their similarities with the query image. Many similarity measures have been developed for image retrieval based on empirical estimates of the distribution of features in recent years. Different *similarity/distance measures* will affect retrieval performances of an image retrieval system significantly.

Euclidean Distance

The euclidean distance is the distance between two points in euclidean space. The two points P and Q in two dimensional euclidean spaces and P with the coordinates (p1, p2), Q with the coordinates (q1, q2). The line segment with the endpoints of P and Q will form the hypotenuse of a right angled triangle.

The distance between two points p and q is defined as the square root of the sum of the squares of the differences between the corresponding coordinates of the points. The two-dimensional euclidean geometry, the euclidean distance between two points a = (ax, ay) and b = (bx, by) is defined as:

$$d(a,b)=\sqrt{(bx-ax)^2+(by-ay)^2} \tag{2}$$

Performance Evaluation

To evaluate the performance of retrieval system, two measurements, namely, *recall* and *precision* , are borrowed from traditional information retrieval[18]. For a query q, the data set of images in the database that are relevant to the query q is denoted as *R*(*q*), and the retrieval result of the query q is denoted as *Q*(*q*). The precision of the retrieval is defined as the fraction of the retrieved images that are indeed relevant for the query:

$$\text{Precision} = \frac{\text{Number of relevant images retrieved}}{\text{Total number of images retrieved}}$$

$$\text{Recall} = \frac{\text{Number of relevant images retrieved}}{\text{Total number of relevant images in database}}$$

$$precision = \frac{|Q(q) \cap R(q)|}{|Q(q)|} \tag{3}$$

The recall is the fraction of relevant images that is returned by the query:

$$recall = \frac{|Q(q) \cap R(q)|}{|R(q)|} \tag{4}$$

3. Objectives

1. This project aims to introduce the problems and challenges concerned with the design and the creation of CBIR systems, which is based on a free Database oriented Image retrieval System.
2. To extract visual content of an image automatically, like color, texture, or shape.
3. With the help of the existing methods, describe a possible solution how to design and implement a task specific descriptor, which can handle the informational gap between a source and images in a Database, making an opportunity for the efficient search hereby.

4. Proposed Work

Relevance feedback is an interactive process that starts with normal CBIR. The user input a query, and then the system extracts the image feature and measure the distance with images in the database. An initial retrieval list is then generated. User can choose the relevant image to further refine the query, and this process can be iterated many times until the user find the desired images. In many cases if we want to search efficiently some data have to be recalled. The human is able to recall visual information more easily using for example the shape of an object, or arrangement of colors and objects. Since human are visual type, we look for images using other images, and follow this approach also at the categorizing. In this case we search using some features of images, and these features are the color, texture and shape.

Our purpose is to develop a content based image retrieval system, which can retrieve using features like color, texture and shape in frequently used databases.

5. Result and Discussion

The figure 3 shows the startup screen of the CBIR system.

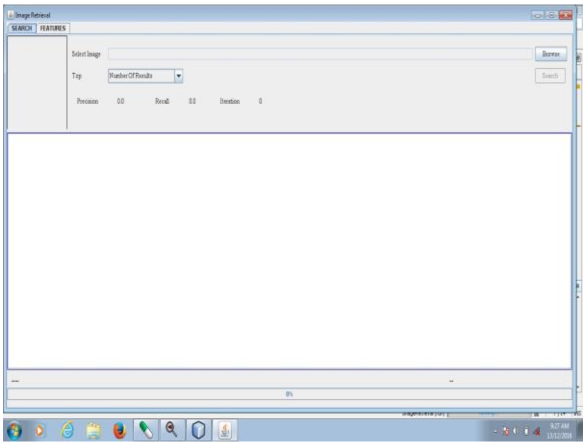


Figure 3: Startup Screen

The next image file shows the feature calculation for images.

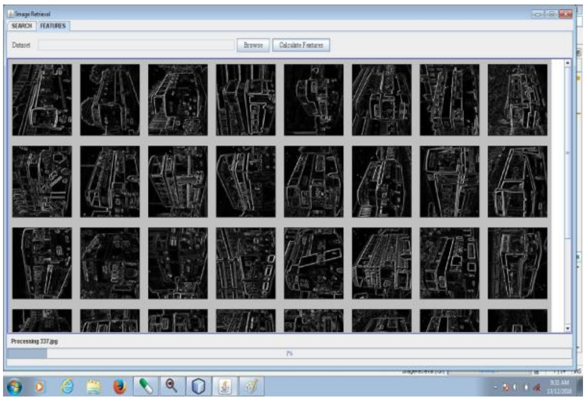


Figure 4: Feature Calculation of Images

The next image files shows searching for images. There are some folders in which images are saved.

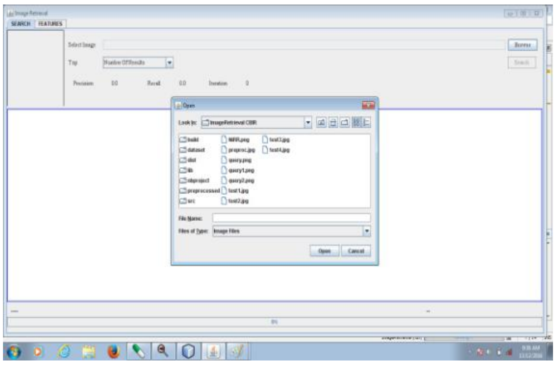


Figure 5: Selection of Image

The next image file shows the processing state of the system.

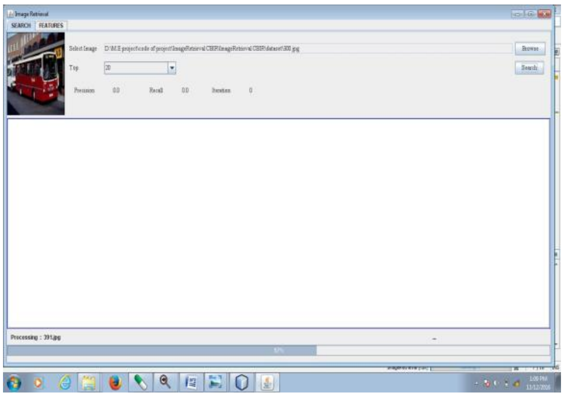
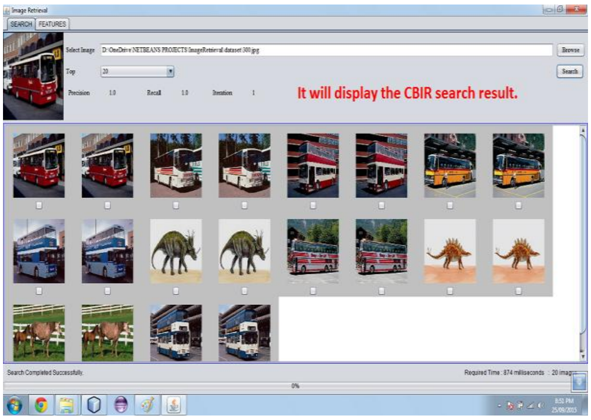


Figure 6 : Processing of the Image Files

The next image file shows the RESULT. This search process continues until the user gets its desired image.



It will display the CBIR search result.

Figure 7: Result (Images are retrieved for given query image)

6. Conclusion

This paper has enabled us to visualize a system that proves to be more accurate in Searching and Extracting Images from databases. Unlike the conventional search methods this technique has enabled us to achieve accuracy in data mining techniques and a feedback system that enables users to get a panoramic view of how and why the specific results were achieved thus making this system more reliable with transparent operations. This paper's implementation in the real time Content based image retrieval systems is also of great significance since a fast accurate technique for systems like Hospital records searches, law enforcement agencies databases and government agencies record will not only improve the standards of service but also make it easier for operators to continue their work.

References

[1] R M Haralick, K Shangmugam, etc. Texture feature for image classification [J].IEEE Transaction on Systems,1973, SMC-3(6): 768-780

[2] Anne H.Solberg S. Jain A K.Texture Fusion and Feature Selection Applied to SAR Imagety[J].IEEE Transaction on Geoscience and Remote Sensing1997.35(2).475-478

[3] B.S. Manjunathi and W.Y. Ma. Texture Features for Browsing and Retrieval of Image Data .IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 18, NO. 8, AUGUST 1996.837-842

[4] A. W. Smeulders, M. M. Worring, A. Gupta, and R. Jain, Content-Based Image Retrieval at the End of the Early Years, IEEE Trans. Pattern Anal. Machine Intell., vol.22 no.12, pp1349-1380, 2000.

[5] Deng, Y., Manjunath, B.S. Unsupervised segmentation of color-texture regions in images and video[J].IEEE Transactions on Pattern Analysis and Machine Intelligence,2001,Vol:23(8),800~ 810

[6] V. Torre and T. A. Poggio. “On edge detection”. IEEE Trans. Pattern Anal. Machine Intell., vol.PAMI-8, no.2, pp. 187-163, Mar. 1986.

[7] E. R. Davies. “Constraints on the design of template masks for edge detection”. Partern Recognition Lett., vol. 4, pp. 11 1-120, Apr. 1986.

[8] W. Frei and C.-C. Chen. “Fast boundary detection: A generalization and a new algorithm ”. lEEE Trans. Comput., vol. C-26, no. 10, pp. 988-998, 1977.

[9] W. E. Grimson and E. C. Hildreth. “Comments on Digital step edges from zero crossings of second Directional derivatives’’. IEEE Trans. Pattern Anal. Machine Intell., vol. PAMI-7, no. 1, pp. 121-129, 1985.

[10] R. M. Haralick. “Digital step edges from zero crossing of the second directional derivatives,” IEEE Trans. Pattern Anal. Machine Intell., vol. PAMI-6, no. 1, pp. 58-68, Jan. 1984.

[11] J. F. Canny. “A computational approach to edge detection”. IEEE Trans. Pattern Anal. Machine Intell., vol. PAMI-8, no. 6, pp. 679-697, 1986.

[12] J. Canny. “Finding edges and lines in image”. Master’s thesis, MIT, 1983.

[13] R. A. Kirsch. “Computer determination of the constituent structure of biomedical images”. Comput. Eiorned. Res., vol. 4, pp. 315-328, 1971.

[14] M. H. Hueckel. “ A local visual operator which recognizes edges and line”. J. ACM, vol. 20, no. 4, pp. 634-647, Oct. 1973.

[15] Rafael C. Gonzalez and Richard E. Woods. “Digital Image Processing”. Pearson Education Inc, 2008.

[16] R M Haralick, K Shangmugam, etc. Texture feature for image classification [J].IEEE Transaction on Systems,1973, SMC-3 (6): 768-780

[17] J. Matthews. “An introduction to edge detection: The sobel edge detector,” Available at http://www.generation5.org/content/2002/im01.asp, 2002.

[18] A. M. W. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years, " *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol.22,No.12, pp. 1349-1380, Dec. 2000.

[19] L. Ballerini, X. Li, R. Fisher, and J. Rees, “query-by-example content-based image retrieval system of non-melonoma skin lesions,” MCBR-CDS’09 Proc. First MICCAI Int. Conf. Med. Content-Based etr. Clin. Decis. Support, vol. 5853, pp. 31-38, 2009