Mathematical Statistician and Engineering Applications
ISSN: 2094-0343
2326-9865

Annotation based on the Text as well as the Incorporation of Visual Aspects for the CBIR System

Sachin Sharma[1], Rahul Bhatt[2], Gesu Thakur[3]

[1, 2]Assistant Professor, Computer Science & Engineering, School of Computer Science & Engineering, Dev Bhoomi Uttarakhand University, Chakrata Road, Manduwala, Naugaon, Uttarakhand 248007

[3]Associate Professor, Computer Science & Engineering, School of Computer Science & Engineering, Dev Bhoomi Uttarakhand University, Chakrata Road, Manduwala, Naugaon, Uttarakhand 248007

[1]socse.sachin@dbuu.ac.in, [2]socse.rahul@dbuu.ac.in, [3]head.ca@dbuu.ac.in

Article Info
Page Number: 1646 - 1663
Publication Issue:
Vol 71 No. 4 (2022)

Abstract

The results of the user's query in a content-based image retrieval system are a group of photos that are ordered according to the feature similarities between the images with regard to the query. In the actual world, photographs serve as the primary source of information and reflect many characteristics of things, such as their color, shape, and other characteristics. We need effective tools for searching, viewing, and retrieving images in a variety of contexts, including the fashion industry, crime prevention, the medical field, and others. Even though a lot of effort has been done to improve the indexing and retrieval of pictures, there is currently no approach that is widely acknowledged for feature extraction, retrieval, and indexing that is both accurate and quick. An algorithm that analyzes a picture based on the objects and RGB components and gives comparable images as the output is provided by us in this work. The photographs are going to be saved in the database with a tag that accurately reflects what they are. These tags may be used for text-based picture retrieval and contribute to the process of extracting visual features. The majority of the currently available search engines rely on tag-based picture retrieval, which is not only inefficient but also less accurate from the user's point of view. In the method that we have provided, we have found solutions to these challenges. It then goes on to describe the experimental findings, which demonstrate how successful the algorithms

are.

Article History
Article Received*: 25 March 2022*
Revised*: 30 April 2022*
Accepted*: 15 June 2022*
Publication*: 19 August 2022*

Keywords: Content-based image retrieval (also known as CBIR), text retrieval, component retrieval, and RGB color retrieval are some of the keywords to keep in mind.

1. Introduction

The size of digital picture collections is growing at a breakneck pace as a direct result of the proliferation of image capture devices, such as image scanners, digital cameras, and the like, which became possible as a direct result of the development of Internet technology. Picture retrieval that is based on the content of the image is very helpful in a variety of applications in the real world. Therefore, as a consequence, we are in need of an extremely effective approach that can automatically extract primary visual aspects from the picture that is being queried and then display the resulting image to the user based on the primary features of the image that is being queried. [1] [2]. The photos are the primary source of information and indicate the characteristics of the items, such as their color, shape, and other characteristics. We need effective tools for searching, viewing, and retrieving images in a variety of contexts, including the fashion industry, crime prevention, the medical field, and others. In order to satisfy the need, a variety of image retrieval systems were created in the past. The goal of content-based image retrieval, or CBIR, is to create a method that can effectively facilitate the searching and browsing of image digital libraries based on automatically generated picture characteristics. This is the goal of the content-based image retrieval project. Using this method, the photographs are cataloged according to their visual content, which includes things like color, texture, and form. [3] The primary goal of a content-based image query is to retrieve images from the database that are comparable to the user's query image. [4] To speed up image retrieval by utilizing simple image features like color, shape, texture, and histograms are recent approaches for it. [5] [5] [5] [5] [5] [5] [5] [5] [5] [5] [5] [5] [5] [5] [5 Even though a lot of effort has been done to improve indexing and retrieval of pictures, there is currently no approach that is widely acknowledged for feature extraction, retrieval, and indexing that is both accurate and quick.

In this research, we introduced an algorithm that can analyze a picture based on the objects that make up the image as well as the RGB components, and it can produce output images that are comparable to the input photos. The photographs are saved in the database along with a tag that provides a description of what they are. These tags may be utilized for text-based picture retrieval, and their inclusion helps to ensure that the extraction of visual information is both quick and accurate. In addition to this, we have provided experimental findings that demonstrate the efficiency of the algorithms that we have devised.

1.1. An Analysis of the Previous Research

The generation of picture information has led to the creation of vast quantities of images as a direct consequence of the growing focus placed on applications that are multimedia in nature. The color histogram is employed in the aforementioned study [4] in order to depict the color information. The histogram of colors was discovered in the HSV color system. The value was quantized into four different bins, while the hue and saturation were each quantized into eight bins. As a color characteristic in the study described in article [5,] a three-dimensional (3D) histogram was utilized. It made use of three different color spaces, notably RGB, IHLS (Improved Hue and Luminance and Saturation), and IHLS (Improved Hue and Luminance and Saturation), although this was of little use for black and white photographs or drawings. In paper [6], feature extraction from query photos, distance measurements, classifier algorithms including neural network classifiers, and the K-nearest neighbor algorithm, as well as performance metrics, were explored.

Extracting the statistical texture characteristics of quantized histograms in DCT blocks using JPEG compressed format pictures is the approach that the author of the article [7] proposes to use in order to get the greatest possible performance when it comes to the retrieval of images. In CBIR systems, a feature is a characteristic that can capture a certain visual property of an image either globally for the entire image or locally for regions or objects, as was discussed in the paper [8]. This visual property can be captured either globally for the entire image or locally for regions or objects. Color, texture, form, and edge are examples of low-level qualities that are often used in CBIR. In publication [9], the authors present an image retrieval system that is going by the name of Wavelet-Based Color Histogram Image Retrieval (WBCHIR). In this approach, the color histogram is employed for determining color features, and the wavelet

representation is used to determine texture. The amount of time required to process the retrieval of an image is cut down as a result of this.

In the study [10], a wide variety of methods are covered. The Edge Pixel Nearby Histogram, also known as EPNH, identifies neighboring pixels, calculates a total based on that information, and then compares that amount to the pictures in the database. However, this approach scans in every direction, making it a time-intensive process. The Histogram of Edge Directions (HED) creates 72 bins, each with a degree value of five degrees. After then, pixel matching and comparisons are carried out at these intervals of five degrees. Therefore, it takes less time, but the accuracy it provides is not much greater than that of EPNH. For the correlation approach, the picture was first cut up into 64 by 64-pixel squares before being indexed, matched, and then compared index by index. However, the level of precision was low, and the amount of time required was substantial. In EHD, picture blocks and sub-blocks are first formed, and then histograms are drawn depending on the intensity of the pixels in five different directions: horizontally, vertically, 135 degrees, 45 degrees, and non-directionally. The standard deviation and skewness of the image's color distribution are two intriguing concepts that are explored in the work [11]. These moments help to define the image's color distribution.

The majority of search engines make use of strategies that are text or tag based to get images. Google's search engine returns results in a relatively short amount of time, yet using it might be time-consuming for the user since it returns a large number of results, which can lead to confusion. Due to the fact that Yahoo employs tag-based image processing rather than content-based processing, sometimes poor performance might be seen. A content-based approach is one way that the procedure may be made more effective. Both a tag-based and a content-based method of picture retrieval are available with Incognita. Image processing that is based on tags is used by Picsearch and Bing as well. Tineye exclusively offers a search engine that is based on the content of images, and users have the option of either uploading a query picture or providing the location of the image they are looking for. Because of this, we have devised a speedy algorithm that does an analysis of a picture based on the objects that make up the image as well as the RGB components, and it produces output images that are comparable to the input photos. In addition, we have provided experimental findings and comparison analysis, both of which demonstrate the efficiency of the suggested algorithms that we have developed**.**

2. The Suggested Architecture for Content-Based Image Searching:

The content of the user's query picture is used by the content-based image retrieval system to determine which photos should be retrieved. A technique that would swiftly retrieve photographs that are similar to others has been suggested by us. Users have the option of using either a text query or an image query to get photos from the database. The accuracy may be improved further by allowing the user to choose the picture type and format. Figure 1 depicts the overall system architecture of the suggested solution**.**

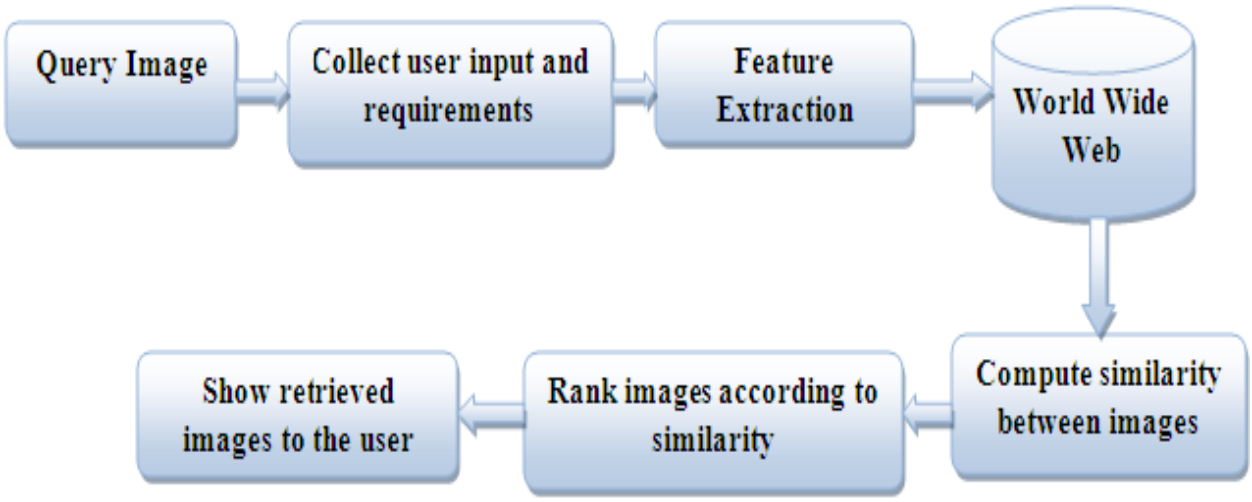

Figure 1. Proposed architecture for content-based image retrieval.

• Query Picture: The user will be prompted to submit the query image when interacting with the user interface in order to get photos of a similar kind.

• Collect the user's input and requirements: The user interface (UI) plays a very essential role in the CBIR system, as it is responsible for collecting the user's intents, needs, and interpretations. The users may access the photographs that correspond to their preferences with the assistance of this component of the system, which is quite beneficial. This method gives the user the ability to select the category of the image as well as the type of image that he wants to retrieve, such as a black-and-white image or a true color image, for example. The user input is taken and based on that, different algorithms can be applied to the query image, as shown in figure 3. In addition, the user can select the category of an image, as shown in figure 2. On the basis of that pick, photographs from the database that have tags that are comparable

may be fetched. These two choices have the potential to make the retrieval both quick and accurate. The whole procedure is seen in figures 2 and 3.

• Feature Extraction: The feature extraction process is the foundational component of the content-based image retrieval system.
When we talk about a feature, we're referring to both visual and specific text-based aspects of it.

• The World Wide Web is a collection of hypertext documents that are connected to one another.
We are able to view web pages via the web browser, and these websites may include text, photos, and other forms of multimedia. When comparing users' images, we may leverage the resources provided by the web thanks to the World Wide Web.

•Calculate the similarity between pictures: The system will, in accordance with the user's query image attributes, compute the similarity between the query image and images already existing on the World Wide Web using a variety of different methods, depending on which option the user chooses.

• Rank photos in accordance with their degree of similarity Once the system has obtained the resulting images from the internet, it will rank the images in decreasing order of the number of features that are matched.

• Display the returned pictures to the user: Once the scanning process is complete, the results of the image retrieval will be shown to the user, along with the opportunity to adjust some visual characteristics of the image, such as the contrast.

Figure 2. Snapshot for taking image category for fast image searching.

3. The following is a proposed algorithm for placing the photographs that have been looked for:

The pictures are retrieved using the many elements of the image, such as the color and qualities of various sections of the image. Figure 3 depicts the user interface that is used for retrieving images depending on their content. The user has the option of entering a query in the form of text or browsing a picture in order to locate more photos that are comparable. The user's requirements, such as picture size and image format, may be filled in according to their preferences. A snapshot of the functions that have been implemented may be seen in figure 3. The photographs from the database that are most comparable to the one you submitted as a query will be retrieved. The picture that is the closest match will appear first in the results. The best possible illustration will be found in Result-1. As a result, the algorithm will place the photos in a ranking that is based on the decreasing number of matching characteristics.

In the implementation, the user will select the type of image that he wants to retrieve, and based on that, one of the two proposed algorithms will be selected. For instance, if the user

wants a black-and-white image to be retrieved, then the component-based analysis algorithm will be selected; this will ensure that the result is extremely accurate and that retrieval will occur quickly. If the user does not pick any kind, we determine the type of query picture and, based on that, choose the method that would provide the most accurate results. In addition to this, we have included the capability to get photos based on text queries.

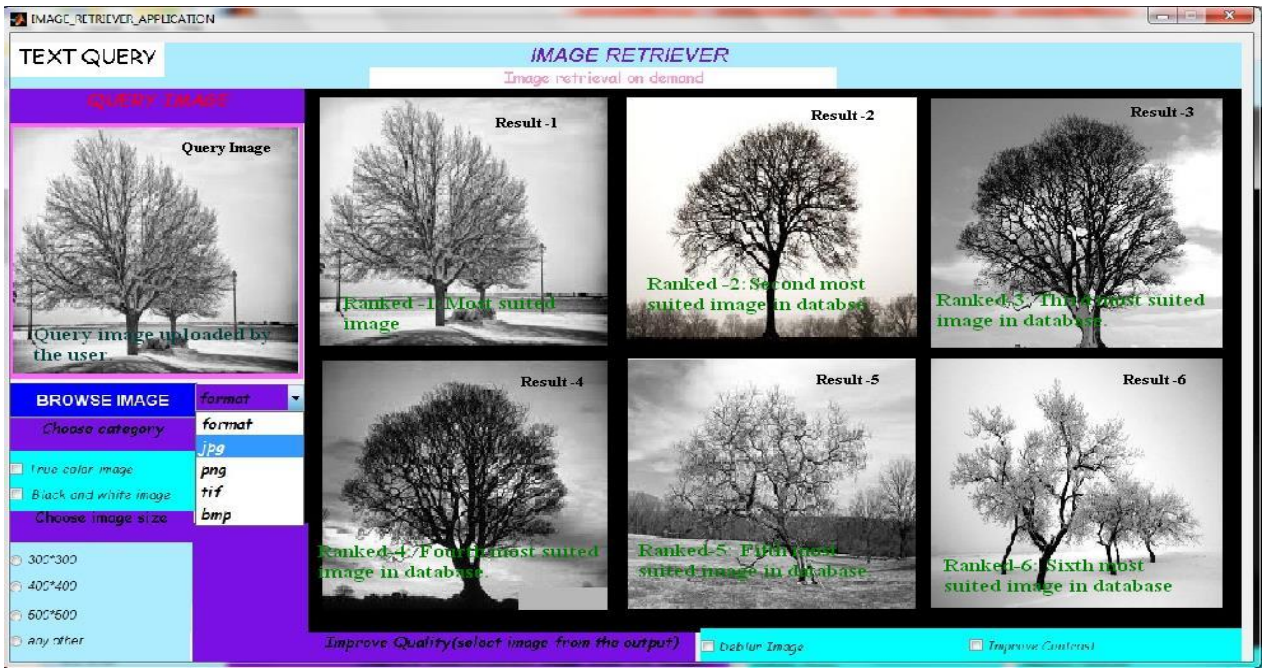

Figure 3. Snapshot of user interface (UI) for content based images retrieved.

Input: User query image (without selecting any category of image type)

Output: Selection of Algorithm to rank images

1.Myimage()

2.{

3.**query** ←retrieve_query_image //read image

4.**Var** ←size(**query**); // returns the size of matrix

5.if number of elements in the size matrix is equal to 3

6.Select RGB component Algorithm

7.else select Component Analysis Algorithm

8. }

Figure 4. Proposed algorithm for finding the type of image and selecting suitable algorithm for ranking.

The different functions used in algorithm (Figure 4) are explained below:

•**Myimage():** In this function we find the type of image that has been input as the query image by the user, based on that we select the suitable algorithm.

•**size():** This function returns size of matrix in separate variables.

Input: User query image

Output: Ranking of images based on components.

```
componentanalysis()
{
filenames ← retrieve_database_images
Query←retrieve_query_image                    //read image
resize query image to n*n
bw←convert the resized query image to binary image
  //component test1
eu = bweuler(bw);                             //obtain euler number of the image

          //component test 2
HQ←process the query image block by block and find hough transformation

  //component test 3
PQ=bwperim(bw);                               //perimemter pixel of image analysis
for each database  image  do {
resize database image to n*n
bw2←convert the resized database image to binary image
eul = bweuler(bw2);
X=Calculatescore(eul,eu);                     //calculate score
```

15. **HS**←process the database image block by block and find hough transformation
16. *X2*=corr2(**HQ**,**HS**);
17. *PS*=bwperim(**bw2**);
18. X_3 =Calculatescore(*PQ*,*PS*);
19. **array**(j)=($X+X_2+X_3$);}
20. sort the **array** in descending order and accordingly place elements in **filename.**
21. Display images of **filenames**
22 }

Figure 5. Proposed component-based analysis algorithm for Ranking of images.

```
1. Calculatescore(a,b)
2. {
3. E=a-b;        //subtract the two parameters
4. X=numel(E)-nnz(E);                    //subtract number of elements in the matrix with total nonzero
5. return((X/numel(E))*100);//returns the percentage difference between the parameters
6. }
```

Figure 6. Proposed algorithm for calculating image similarity score based on component analysis.

- **component analysis() (figure 5.):** The Proposed component-based analysis algorithm for Ranking of images is shown in figure 5. This function analyzes the image based on components present in it. In beginning it resizes the query image to any n*n size preferably 300*300, then the image if not already black & white is converted to a binary image and on that image 3 tests are done considering various parameters. In first test Euler number is calculated, in second test SHT based analysis is done where image is processed in smaller blocks and in third test perimeter pixels are analyzed. In a similar way the database images are analyzed and three parameters are obtained. We send the parameters to Calculatescore() function to get a score of image similarity ,these scores are added and placed in an array.

The array after comparing all database images with query image is sorted and according to it the array containing image list is sorted and in this way get ranked. Finally the ranked images are displayed.

- **Calculate score() (figure 6.):**Proposed algorithm for calculating image similarity score based on component analysis is shown in figure 6. It finds the difference between two input parameters and a percentage of it is calculated which is treated as a score of the image similarity in the componentanalysis() function.

Input: User images query.

Output: Ranking of images based on RGB component.

1. el=input('Select any image name:','s'); //image selected for
2. **MyImage**=imread(image_sel);
3.**MyImage2**= resize query image to 300*300
4.**matstr2**=**matstr**; //copies **matstr** list to **matstr2**
5.**s**=imread(char(**matstr2**(1))); //initializing s with first image matrix
6.**s1**= resize image s to 300*300
7.for each image do {
8.**s**=imread(char(**matstr2**(*m*))); // read image from list of images in database
9.**s1**= resize image s to 300*300
10.**J**=**MyImage2-s1**;
11.**str**='';*count*=0;*i*=1;
12.images_string_array();
13.sort_string()
14.sort_string()
15.for *i*=1 to total number of image in database do {
16.**figure**(*i*);imshow(char(**matstr2**(*i*))); //show closely related images to our query image
17.}

Figure 7. Proposed algorithm for Ranking of images based on RGB component

Input: J image matrix with dimensions 300x300 px

Output: str2 - An array of strings.

```
1) images_string_array(){
2) divide each image matrix viz J into 60x60 px blocks
3) i=0
4) for each image I in database do {
5) str(i ,:)=''                    // str(i,:) matrix of strings with string at i^th  row
6) for each 60x60 px blocks in an  image I do {
7) count=  no. of non zero in each block
8) if count>8100  str(i , :)=str(i , :) + 'a'
9) else if count>5400  str(i , :)=str(i , :) + 'b'
10)        else if count>5400  str(i , :)=str(i , :) + 'c'
11)        else str(i , :)=str(i , :) + 'd'
12)        }}
```

Figure 8. Proposed algorithm for dividing image matrices into blocks and providing each image a string value.

A list of photos is at our disposal. Take a picture as input from the user and refer to it as the query image. Adjust the size of the picture matrix so that it is 300 by 300. Resize each picture in the database to 300 by 300 pixels by selecting it one at a time from the database. Consider the differences between this picture and the one you Query provided. The result is a new matrix denoted by the letter J, as seen in figure 7.

• images string array(): Figure 8 presents the proposed technique for breaking picture matrices into blocks and assigning a string value to each image. This algorithm is used by the images string array() function. This picture matrix is split up into smaller matrices that are 64 pixels by 64 pixels. We provide each block a character constant that serves as its identifier. Blocks that have between 60 and 80% zeros will be given the letter 'b,' blocks that have between 30 and 60% of their pixels set to zero will be given the letter

'd.' Therefore, each picture has 25 blocks, which implies that each image is a string that contains 25 characters. An array of strings serves as the storage format for this information.

Input: String array consists of information of density of zeroes in blocks.

Output: Sorted list of images '**matstr2**'.

1) sort_string() {
2) For *i*=1 to total no.of image in database do {
3) count total no. of character a,b,c,d in each string i.e. str2(i,:)
4) }
5) rearrange list of images in matstr2 such that, // can be done with any simple way

image having largest no. of a's, comes at top and
if two images have same no. of a's only then we check for greater no. of b's.
if two images have same no. of b's only then we check for greater no. of c's.
if two images have same no. of c's only then we check for greater no. of d's.

Figure 9. Proposed algorithm for ordering images according to density of zeros in a block.

• sort string(): figure 9 illustrates the proposed approach for arranging pictures based on the density of zeros in a block. this algorithm is used by the sort string() function. This array of strings is sorted in such a way that the picture with the highest number of a's in its string is placed at the top of the array. If two photos have the same number of a's alone, we check to see which one has a higher number of b's. The image that has a higher total number of b will be shown on top among them. If the numbers of b in both matrices are the same, the next step is to compare the numbers of c and d, and vice versa if the numbers of c in both pictures are the same. As a result, we get a list of photos that, starting at the top of the list, offers us images that are highly relevant to our Query Image.

4. The results of the experiments and an examination of the algorithms:

In this part, we present the outcomes of our experiments, which demonstrate how successful our suggested algorithms are. During the testing of the component analysis method, we utilized 50

photos of trees and 50 photographs of birds.

We selected black-and-white photographs, ran four separate tests with various query images, and determined the amount of time it took for the results to be presented. Figures 3 and 10 show the photos that were produced as a result. The picture that is most relevant to the one you submitted as a query is going to be Result-1. Consequently, the algorithm will place the photos in a ranking that is based on the decreasing number of features that are matched. The time required to retrieve an item is shown in tables 1 and 2.

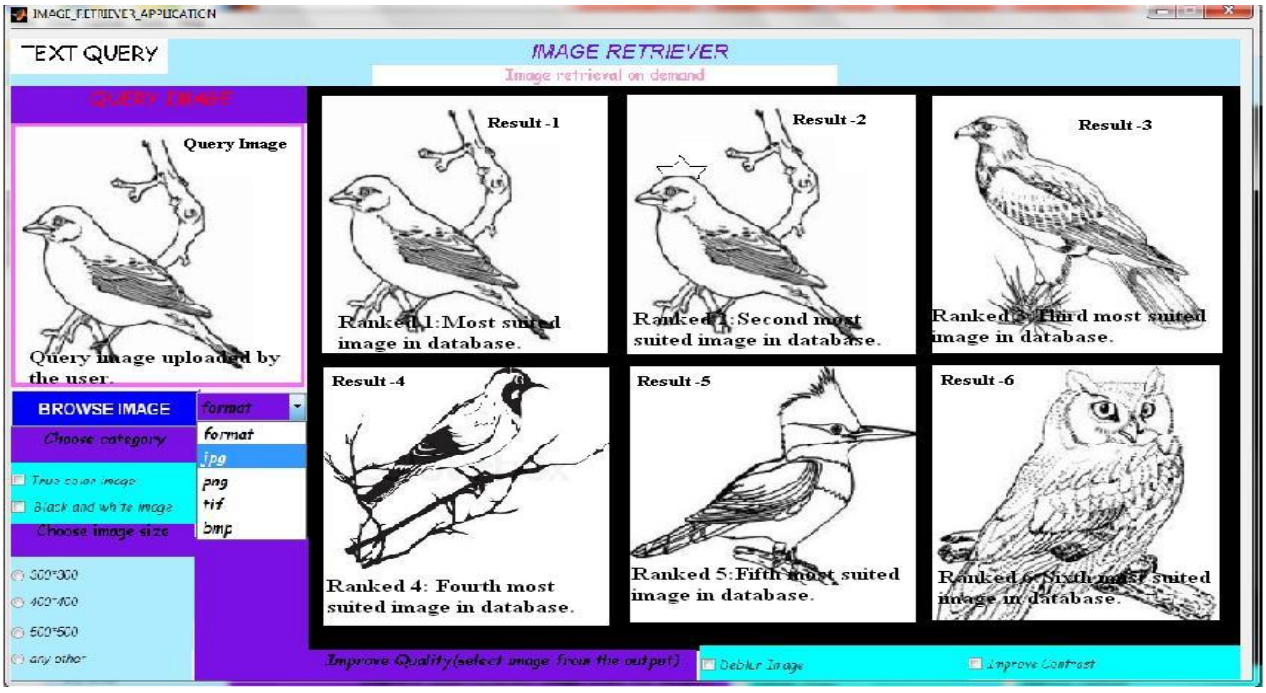

Figure 10.Bird database

Table1: An example of a table: Output time for ranking of images obtained (without tag extraction) based on component analysis.

Bird Database (total 50 images)		**Tree Database (total 50 images)**	
Test Number	***Time (in sec)***	***Test Number***	***Time (in sec)***

Test 1	1.051326	*Test 1*	1.023126
Test 2	1.009962	*Test 2*	1.121006
Test 3	1.001326	*Test 3*	1.000106
Test 4	1.008208	*Test 4*	1.054286

Table 2: Output Result's time for ranking of images obtained (with tag extraction) based on component analysis.

Bird and Tree Database (total 100 images)	
Test Number	***Time (in sec)***
Test 1	2.123410
Test 2	2.233316
Test 3	2.144542
Test 4	2.186632

We have used 100 images of cars in test of RGB component analysis algorithm (Figure.11). We used true color images and conducted four successive tests with different query images and calculated the time in which results were displayed.

Table 3: Output time for ranking of images obtained (with tag extraction) based on RGB component.

Car Database (total 100 images)	
Test Number	***Time (in sec)***
Test 1	2.110111
Test 2	2.200116

Test 3	2.101942
Test 4	2.112332

Table 4. Comparison with common retrieval systems

UNIVERSE DATABASE(500 images)					
NAME	***total time (s)***	***REMARK (for time)***	***Precision***	***REMARK (for precision)***	***Overall***
SIFT	*20.156658*	*bad*	87%	*good*	Good robust system
SURF	*5.86*	*best*	67.8%	*bad*	Better in speed
OUR SYSTEM	*7.1100*	*good*	92%	*best*	Use of two algorithms makes the system accurate and fast.

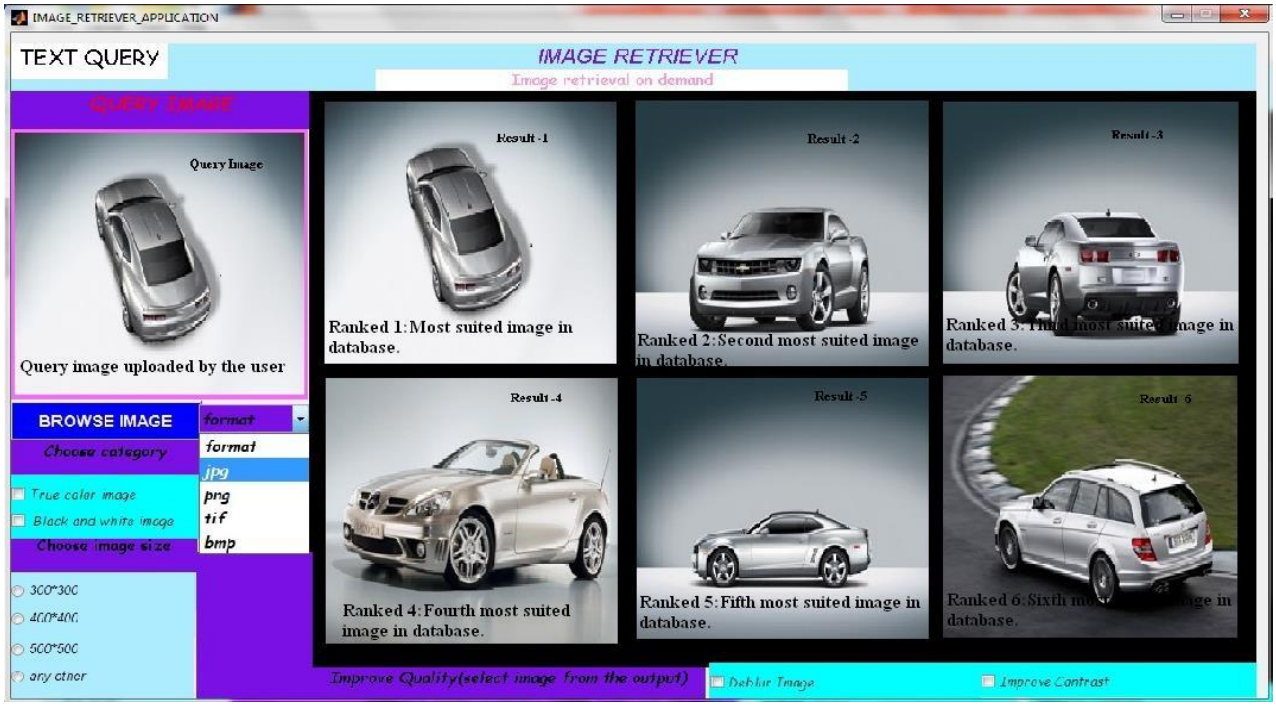

Figure 11. car database

Conclusion

In this paper, we have proposed a content-based image retrieval system that has better functionality and also improves accuracy when compared to some common image retrieval systems that are in use these days. Additionally, this system improves the speed at which images can be retrieved from the database. Combining a number of different algorithms is the strategy that will be used in order to get superior outcomes in a content-based picture retrieval system. The photographs are saved in the database along with a tag that provides a description of what they are. These tags may be utilized for text-based picture retrieval, and their inclusion helps to ensure that the extraction of visual information is both quick and accurate.

References:

[1] Shaoting Zhang, Junzhou Huang, Hongsheng Li, and Dimitris N. Metaxas, Senior Member, IEEE, "Automatic Image Annotation and Retrieval Using Group Sparsity", IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B: CYBERNETICS, VOL. 42, NO. 3, JUNE 2012

[2] Xinning Shen, Xiaolong Wang, Jianhong Du, "A Novel Image Retrieval Method Based on ColorAutocorrelogram and Mutual Information", International Conference on Advanced Computer Science and Electronics Information (ICACSEI 2013).

[3] R.Venkata Ramana Chary, D.Rajya Lakshmi Gitam, K.V.N Sunitha, "Color Image Techniques for Image Retrieval in Large Image Set of Database", IJCSNS International Journal of Computer Science and Network Security, VOL.13 No.1, January 2013.

[4] Lining Zhang, Lipo Wang andWeisi Lin. Generalized Biased Discriminant Analysis for Content-Based Image Retrieval. IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics, Vol. 42, No. 1, pp. 282-290, 2012.

[5] Imtnan-Ul-Haque Qazi, OlivierAlata, Jean-ChristopheBurie, Ahmed Moussa, ChristineFernandez-Maloigne. Choice of a pertinent color space for color texture characterization using parametric spectral analysis. Pattern Recognition 44, pp. 16–31, 2011

[6] H.B. Kekre, Dhirendra Mishra, Anirudh Kariwala. A Survey of CBIR Techniques and Semantics. International Journal of Engineering Science and Technology (IJEST), Vol. 3, No. 5, PP. 4510-4517, 2011.

[7]Fazal-e-Malik, Baharum Bin Baharudin and KifayatUllah."Efficient Image Retrieval Based on

QuantizedHistogram Texture Features in DCT Domain"Frontiers of Information Technology, 2011

[8] Selvarajah S and Kodituwakku S. R. Analysis and Comparison of Texture Features for Content Based Image Retrieval. International Journal of Latest Trends in Computing, Vol.2, Issue 1, pp. 108 – 113, 2011.

[9]Manimala Singha and K.Hemachandran .Content Based Image Retrieval using Color and Texture Signal & Image Processing : An International Journal (SIPIJ) Vol.3, No.1, February 2012

[10] Adel Hafiane et al. "Region-based CBIR in GIS with local space filling curves to spatial representation",(2006), Elsevier Pattern Recognition Letters 27 (2006) 259–267.

[11] Hatice Cinar Akakin and Metin N. Gurcan. Content-Based Microscopic Image Retrieval System for Multi- image queries. IEEE Transactions on Information Technology in Biomedicine, Vol. 16, No. 4, pp. 758 – 769, 2012.