12-2016

Building an Efficient Content Based Image Retrieval System by Changing the Database Structure

Ishan P. Ranasinghe Arachchilage Mr.
St. Cloud State University, ishan461@yahoo.com

Building an Efficient Content Based Image Retrieval System by Changing the Database Structure

by

Ishan Ranasinghe Arachchilage

A Thesis

Submitted to the Graduate Faculty of

St. Cloud State University

in Partial Fulfillment of the Requirements

for the Degree

Master of Science in

Computer Science

December, 2016

Thesis Committee:
Jayantha Herath, Chairperson
Ezzat Kirmani
Susantha Herath

Abstract

Content Based Image Retrieval (CBIR) is still a major research area due to its complexity and the growth of the image databases. Color Based Image Retrieval is one of the major retrieval methods in Content Based Image Retrieval systems. At present, researchers combine image retrieval techniques to get more accurate results. With the large image databases, image retrieval is still a challenging area and the efficiency of the image retrieval techniques still need to be considered. For this purpose, a comparative study of image retrieval techniques has been discussed in this paper. In addition, an efficient method is presented which aids to retrieve images by storing an intermediate result of the process in the database. To compare the query image and the images in the database, Euclidean distance, Normalized Cross Correlation distance and Histogram Intersection distance are taken as distance measures. Experimental results demonstrate Histogram Intersection distance is better than the other two methods. The intermediate result was stored using an event in the system. By making minor modifications to the proposed system, it creates a possibility for the user to add images to the database just by clicking on a button. Thus, the user can expand his/her database on his/her own will. Results show a significant improvement of performance in the proposed method.

Table of Contents

Chapter Page

List of Tables

List of Figures

Figure Page

Chapter 1: INTRODUCTION

Overview

With the growth of World Wide Web, interest in the digital images has increased immensely. Digital images are being used for many fields such as medicine, designing, journalism, education, etc. There are a variety of methods and techniques that could be used for storing and retrieving images. However, most of these engines depend on Meta data (keywords, tags, descriptions). These engines are inefficient, costly and may not capture every keyword or tag that is given for the image. Nevertheless, if a system can filter images based on their contents it could deliver results that are more accurate.

Content Based Image Retrieval System has been introduced as a solution for the conventional image retrieval systems. The conventional image retrieval systems are known as Keyboard Based Image Retrieval (KBIR) Systems. In addition to those two retrieval systems, Semantic Based Image Retrieval (SBIR) systems are also available today [1]. Even Google and Bing search engines utilize this method. Semantic search engines are capable of retrieving images by considering current trends, the location of the search, intend of the search, variations of the words in semantic search, etc. [2].

Content Based Image Retrieval is a process to retrieve a stored image from a database by supplying an image as a query instead of text [3]. It is used for, but not limited to applications such as facial recognition systems, medical diagnosis, architectural and engineering and information systems. The effective content based image retrieval needs efficient extraction of low-level features like color, texture and shapes for fast query image matching as well as for retrieval of similar images [4]. Due to the benefits of content based

image retrieval engines, it is important to improve the efficiency of feature extraction techniques.

Problem Statement

Most of the CBIR systems focus on accuracy. Modern researches combine content based image retrieval techniques combined to develop content based image retrieval systems. For example, in 2016 A. Anandh, Dr. K. Mala, and S. Suganya proposed a CBIR technique using multiple feature extraction techniques to enhance the accuracy of the image retrieval process. Color Auto-Correlogram Feature, Gabor Wavelet Feature and Wavelet Transform feature combined to build content based image retrieval system. Results clearly show the improvement of the accuracy [5]. Due to computational cost, with the improvement of the accuracy, it is possible to decrease the retrieval speed of the systems with multiple feature extraction techniques. It is important to improve the performance of the system as it helps improve the accuracy of the system.

Proposed Solution

In order to enhance the performance by retrieval speed, some changes have been implemented to the database. An intermediate result of the image comparison has been stored in the database. The results show a significant performance enhancement with the proposed system.

Report Outline

This thesis report is divided into five sections. Introduction, Related Work, Methodology, Experimental Results and Discussions and Conclusions and Future Works. In chapter 1(Introduction) an overview of content based image retrieval systems, problem statement, proposed solution and report outline were discussed. In Chapter 2 (Related Work)

existing content based image retrieval techniques are explained. Comparison between RGB model and HSV model, how to convert an RGB model to HSV model, Image comparison techniques, the analysis in distance metrics, Content based image retrieval based on neural networks and performance evaluation are discussed. The methodology of the developed content based image retrieval system is explained in Chapter 3 (Methodology). Used image comparison algorithms and image retrieval algorithms are explained in this chapter. Database structure, algorithm and implementation is included. Due to the lengthy time consumption in image retrieval time, it is essential to make improvements to the performance of the system. The built system was improved by changing the database structure and utilizing an image retrieval technique. Chapter 3 also discusses the database structure, algorithm and implementation of the proposed system in detail. After retrieving distance measures, results were analyzed and plotted graphs were created to identify the best distance measure to retrieve images. In addition, the proposed method was compared with the previous version of the content based image retrieval system. The analyzation, results, and discussions regarding the comparison are included in Chapter 4 (Experimental Results and Discussions). Finally, in Chapter 5 (Conclusion and Future Work) an evaluation of the various techniques proposes a need to build a content based image retrieval system.

Chapter 2: RELATED WORK

Comparison between Color Model

In Content Based Image Retrieval Systems information can be extracted based on many features. Color, texture, and shape are the most commonly used features in these systems. It is possible to increase the accuracy by considering multiple features in the information extraction process.

RGB and HSV are two color models that can be used to implement a CBIR system. Figure 1 shows the two color models [6].

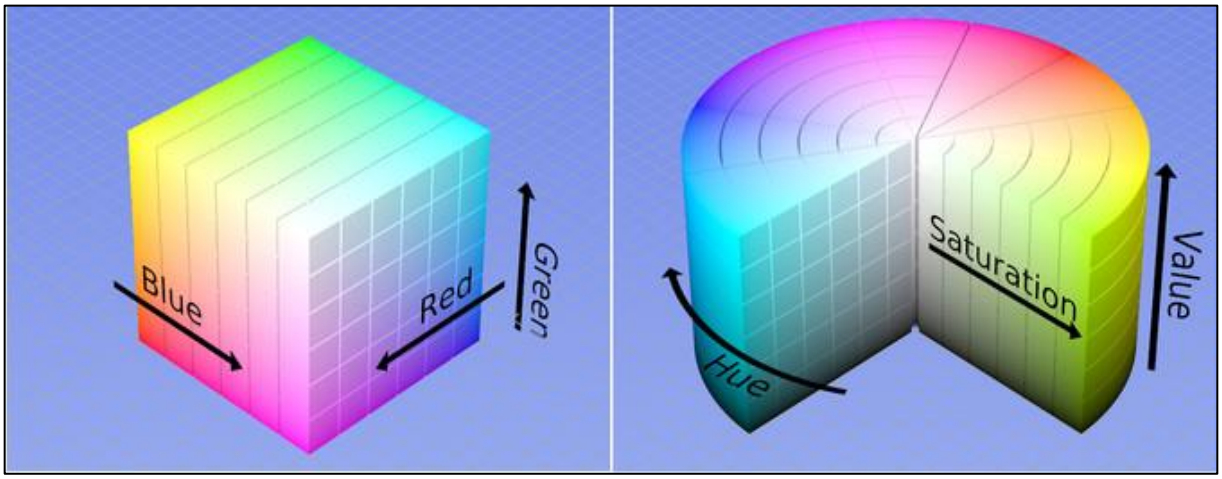

Figure 1: RGB vs. HSV color model.

The R, G, B represent red, green and blue components respectively with values between 0-255. HSV stands for Hue, Saturation, and Value. The Hue color type ranges from 0 to 360, Saturation ranges from 0 to 100% and Value ranges from 0 to 100%. The following formula demonstrates that it is possible to convert images from one color model to another [7].

$$H = \cos^{-1}\frac{\frac{1}{2}[(R-G)+(R-B)]}{\sqrt{(R-G)^2+(R-B)(G-B)}}$$

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

$$V = \frac{1}{3}(R+G+B)$$

One drawback with the RGB model is its behavior when the illumination in an image changes. With the illumination, the distribution of RGB values will change proportionally, hence giving a very different histogram.

Image Comparison Techniques

Color based. In 2011, a paper authored by N. Sharma, S. P. Rawat, and J. Singh, "Efficient CBIR using Color Histogram Processing", used a content based image retrieval method based on the color histogram approach. When computing a color histogram for an image, the different color axes are divided into a number so-called bins. A three-dimensional 256x256x256 RGB histogram would, therefore, contain a total of 16777216 such bins. When indexing the image, the color of each pixel is found, and the corresponding bin's count is incremented by one. For each R G B, they created separate histograms. For comparing images, they used cross bin distance.

$$L1 = \sum_{i=1}^{n}(Q_i - I_i)$$

Where Q_i is the value of bin i in the query image and I_i is the corresponding bin in the database image [8].

Texture based. Textures are visible patterns which cannot exist as of a single color or intensity. To feature extraction in content based image retrieval systems, valuable information regarding the structural arrangement of background and the surface, texture of an image can be used. Tamura texture feature, Steerable Pyramid, Wavelet Transform, Gabor Wavelet Transform are some of the common feature extraction techniques based on texture [9].

Many content based image retrieval engines use Gabor Wavelet Transform (GWT) to extract features based on texture. Let G_{mn} (x, y) be the Gabor filter applied to the original image I(x, y) of size PxQ. Then magnitudes E (m, n) can be written as [5],

$$E(m,n) = \sum_{m} \sum_{n} |G_{mn}(x,y)|$$

Where, m=0, 1,….,M-1 and n=0,1,….,N-1

To identify images with similar texture, mean and standard deviation should compute from magnitude array.

$$\mu_{mn} = \frac{E(m,n)}{P * Q}$$

$$\sigma_{mn} = \frac{\sqrt{\sum_x \sum_y (|G_{mn}(x,y)| - \mu_{mn})^2}}{P * Q}$$

Jigisha Patel and Nikunj Gamit compared texture feature extraction techniques in their IEEE WISPNET 2016 conference paper. Table 1 shows a comparison of texture feature techniques [9].

Table 1

Comparison in Texture Feature Techniques

Texture Feature Techniques	Advantages	Disadvantages
Tamura Feature	Generated the histogram for texture features	Represent the specific types of texture
Steerable pyramid	Rotation invariant	More computation and storage
Wavelet Transformation	Less retrieval time	Poor performance compared to the Gabor wavelet transform
Gabor Wavelet Transform	Achieves highest retrieval results	Computationally intensive

Shape based. In both color based image retrieval approach and texture based image retrieval approach, the object in the image is not taken into consideration. In order to retrieve images accurately, the shape of the objects should also be taken into consideration. S. Deniziak and T. Michno proposed a method to retrieve images based on the shape of the objects. Their image retrieval system is based on image decomposition into primitives with attributes [1]. Figure 2 shows the methodology proposed by Deniziak and Michno.

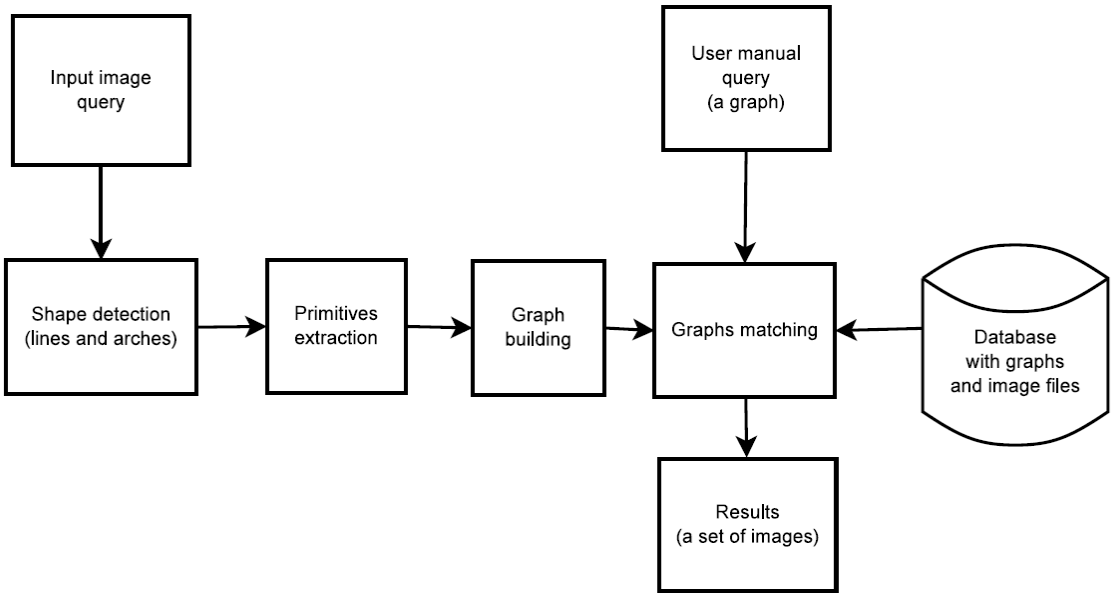

Figure 2. Shape based image retrieval system.

Analysis in Distance Metrics in Content Based Image Retrieval

There are several methods to retrieve images from a database based on its content. In the retrieving process to measure the similarity between images a distance metric can be used. Researchers are still trying to identify the best distance metric to retrieve images in image retrieval system. Euclidian distance, Manhattan distance, and Vector Cosine Angle Distance are a few of the distance measures that can be used to develop Image retrieval system [10].

Euclidian distance. If u = (x_1, y_1) and v = (x_2, y_2) are two points, then the Euclidean distance between u and v is given by,

$$EU(u,v) = \sqrt{(x1 - x2)^2 + (y1 - y2)^2}$$

Manhattan distance. If u = (x_1, y_1) and v = (x_2, y_2) are two points, then the Manhattan Distance between u and v is given by,

$$MH(u,v) = |x1 - x2| + |y1 - y2|$$

Vector cosine angle distance. If we consider two vectors X and Y where X ≡ (x_1, x_2,…., x_n) and Y ≡ (y_1, y_2,…., y_n), then cosθ may be considered as the Cosine of the vector angle between X and Y in n dimension. Formally, we define VCAD as follows,

$$VCAD(X,Y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i {x_i}^2}\ \sqrt{\sum_i {y_i}^2}}$$

Normalized cross correlation. Cross Correlation is simply the measure of similarity between two signals. The sum of pairwise multiplications of corresponding pixel values of the images can be taken as the cross-correlation. Image correlation can be defined as the numeric measure of image similarity [11].

$$Cross - Correlation(Image1, Image2) = \sum_{x,y} Image1(x,y) \times Image2(x,y)$$

There is also an enhanced version of cross-correlation method. It is known as the Normalized cross-correlation method [11].

$$NCC(Image1, Image2) = \frac{1}{N\sigma_1\sigma_2}\sum_{x,y}(Image1(x,y) - \overline{Image1}) \times (Image2(x,y) - \overline{Image2})$$

Histogram intersection distance. Histogram Intersection Distance can be written as,

$$S(A,B) = \frac{\sum_{i=1}^{n} \min(Hi(A), Hi(B))}{\sum_{i=1}^{n} Hi(A)}$$

Where A and B represent two histograms [12].

Content Based Image Retrieval using Neural Network

Content based image retrieval systems based on color, texture, and shape cannot search images based on their semantic content. To resolve this issue a neural network can be used. There are many ways a researcher can build a neural network to retrieve images. In this paper, one method is discussed using a conference article.

H. Karamti, M. Tmar, and F. Gargouri proposed a content based image retrieval system in 2014, using a neural network. They used back propagation algorithm to train the network. As the first step, they extract low-level features (color, texture, and shape) j = 1, 2…m from images in the database and creates feature vectors [$F_i = F_{i1}, F_{i2}….F_{im}$] for each image i. Then they create extract features from a query image and evaluate the similarity measure using Euclidean distance. A score vector [$S_i = S_{i1}, S_{i2}….S_{in}$] was built using query results from the similarity scores where n is the number of images in the database.

To construct the neural network set of queries were taken $q_1, q_2…q_n$. A feature vector $F_{qi} = [F_{qi1}, F_{qi2}….F_{qim}]$ was created for each query (where F_{qij} is the value of the feature in the query qi). A score vector $S_{qi} = S_{qi1}, S_{qi2}…S_{qin}$ generates by image retrieval process (where S_{qij} is the value of similarity score between query qi and image j). Matrix W characterized by this model.

$$F_{qi} \times W = S_{qi}$$

In their research paper, using a set of F_{qi} values as an input layer (L_{input}) and by taking S_{qi} as the output layer (L_{output}) they attempt to predict matrix W [13]. Figure 3 shows the proposed neural network model by Hanen, Mohamed, and Faiez.

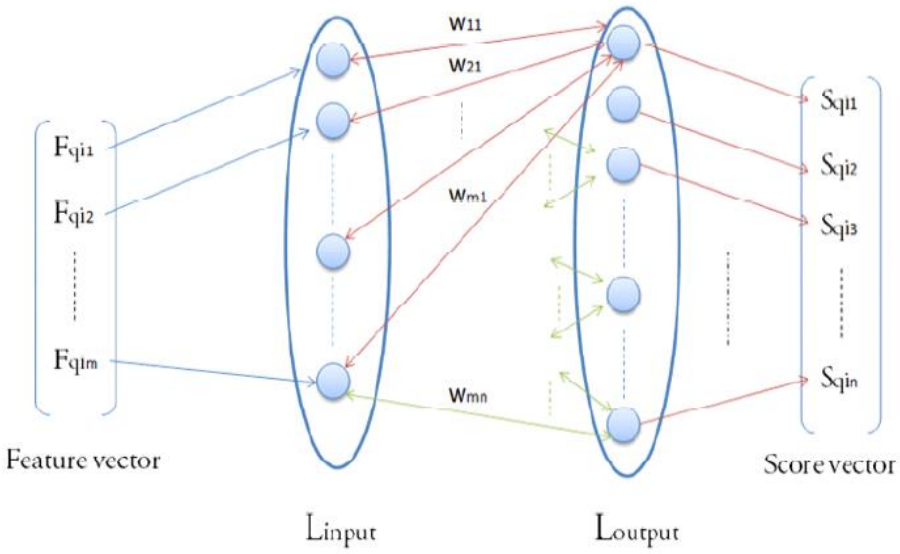

Figure 3. Proposed neural network model by Hanen, Mohamed, and Faiez

$$\begin{pmatrix} F_{q_i 1} \\ F_{q_i 2} \\ \vdots \\ F_{q_i m} \end{pmatrix} \times \begin{pmatrix} w_{11} & w_{12} & \dots & w_{1m} \\ w_{21} & w_{22} & \dots & w_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n1} & w_{n2} & \dots & w_{nm} \end{pmatrix} \cong \begin{pmatrix} S_{q_i 1} \\ S_{q_i 2} \\ \vdots \\ S_{q_i n} \end{pmatrix}$$

Initially w_{ij} values are some random values and then they calibrated in the learning process. The learning process will be repeated until the error of the validation set reduces. Once the W matrix is constructed, images can be retrieved without finding distance measures.

Performance Evaluation in Content Based Image Retrieval System

Content based image retrieval system should be tested for both performance and accuracy. Measuring accuracy is a large issue since it can be only measured by feedback from the people. To obtain more accurate results it is important to use Relevance Feedback technique for content based image retrieval system.

In Relevance Feedback (RF) Techniques, an existing query to retrieve images repeatedly and automatically altering to information is fed back by the user. The user delivers the feedback as to whether the outcomes are relevant or non- relevant after obtaining results. The feedback loop is repeated multiple times if the outcomes are not-relevant [14].

Precision and recall are the most common evaluation measures used in IR [15].

$$Precision = \frac{No.relevant\ documents\ retrieved}{Total\ No.documents\ retrieved}$$

$$Recall = \frac{No.relevant\ documents\ \ retrieved}{Total\ No.relevant\ documents\ in\ the\ collection}$$

$$Error\ Rate = \frac{No.\ \ non-relevant\ images\ retrieved}{Total\ No.images\ retrieved}$$

Chapter 3: METHODOLOGY

The main purpose of this research is to implement an efficient content based image retrieval system. A method has been discussed in this thesis to store histogram in a database to retrieve images much quicker than previously possible. Multiple research studies have already been conducted to compare image matching techniques for content based image retrieval system. Three distance measures were taken to find the best distance measure for the image retrieval system and used the results to develop the content based image retrieval system.

Method used to compare existing content based image retrieval techniques is shown in Figure 4.

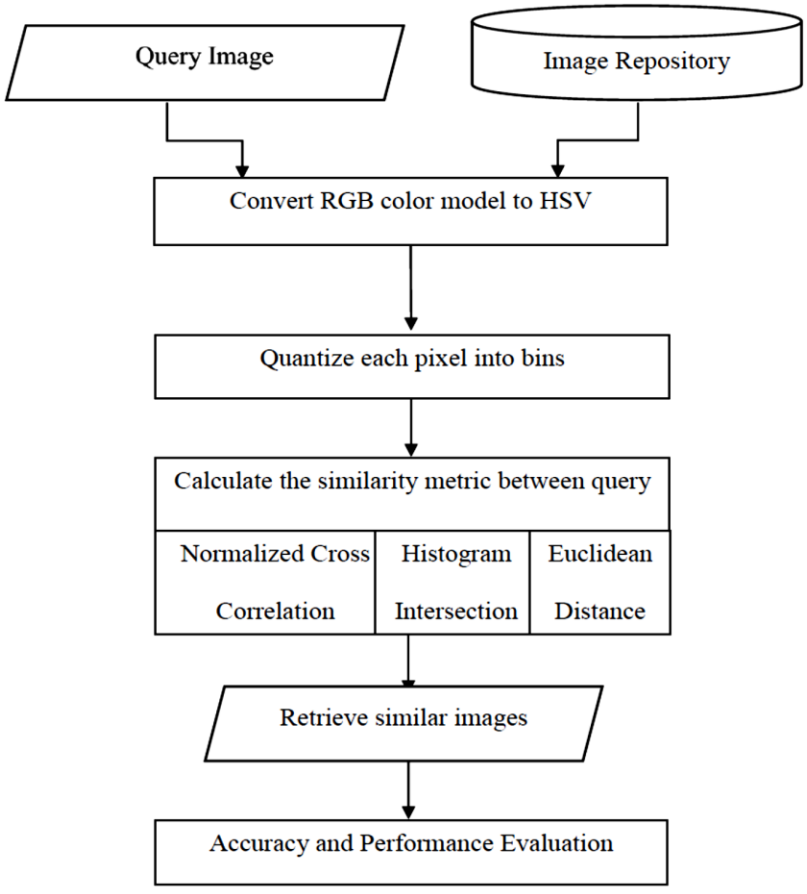

Figure 4. Distance measures comparison and image retrieval.

Image Comparison

Images are originally in RGB color model. However since RGB values change proportionally to illumination, it may not provide accurate results in content based image retrieval systems. Thus in this content based image retrieval system, as the first step both user entered image, and the images in the database are converted to the HSV model. To draw the histogram each pixel in HSV color images are quantized to bins. Quantizing pixels into a large number of bins allows generating more accurate results. On the other hand, it causes slowdowns in the process. Researches who focus on giving quicker results quantized pixels into a small number of bins. Most of them used 36 bins to develop their system. However, in this system, the focus is more on accuracy and therefore pixels are quantized to 160(10*4*4) bins. “Hue” is divided into 10 parts, “Saturation” is divided into 4 parts and “Value” divided into 4 parts [16] [17] [18]. Figure 5 shows the separation of bins in HSV color space.

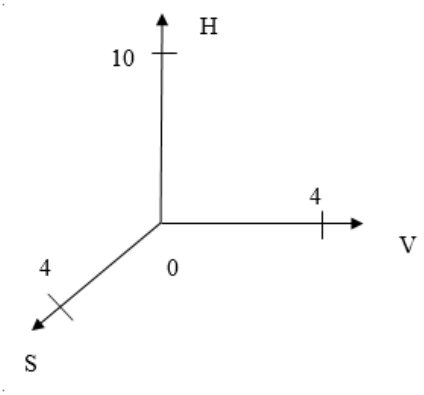

Figure 5. Quantizing HSV color model pixels to bins.

After quantizing pixels to bins similarity metric between user entered image and images from the database has been calculated. Euclidean distance, Histogram intersection distance, and Normalized Cross correlation distance have been utilized as distance measures for this system.

Image Retrieval

In this method, an image is taken from the user and compared to each image in the database. This process is typically time-consuming and this report also discusses a solution for this issue. After taking an image from the user, the system connects to the image database. The database consists of an image identification number as well as a path to each image. Image Database (Image_Identification_Number, Image_Path).

To retrieve these values to the program, a 2D array was created and the image identification number and the image path were stored on it. By taking one image at a time (using image path in the 2D array), the distance between the image taken and the user entered image would be calculated. After taking distance, the distances are sorted and the first nine images with the smallest distances are taken. By cross-referencing taken images and 2D array which has image paths, the images were retrieved. Figure 6 shows cross-referencing of image table and distance array.

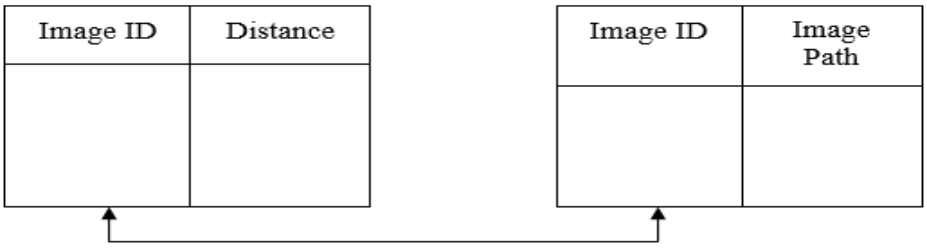

Figure 6. Cross-referencing of image table and distance array.

Result Analyzing

Retrieved images from each distance measure have been analyzed in this section. For selected images in each distant measure, a scatter plot was created to help analyze how distance distributes. The ranges of the distance values are different for each distance method.

Thus it is necessary to standardize distance to analyze the data. For standardization, the distance values “Min-Max normalization” were used [19].

$$new_{value} = \frac{value - min_x}{max_x - min_x} \times (D - C) + C$$

Where, [C, D] are predefined boundaries. To implement this system C is taken as 0 and D is taken as 100. Therefore the value can be taken as a percentage.

Implementation

The system was developed using C Sharp programming language. MySQL database management system was used to create the database. A separate image database with only 27 images was created to test the image comparison method.

Database. Figure 7 shows the database structure of the first version of content based image retrieval system.

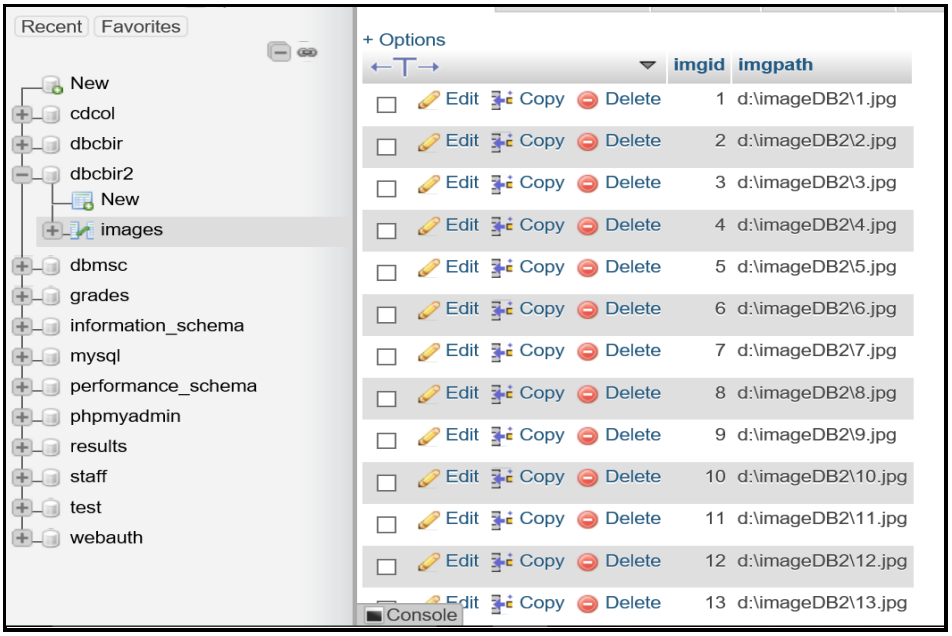

Figure 7. Image database.

Algorithm.

Connect to Database
Get all the records in Image table to Reader object
Generate histogram to user entered Image(Histogram values stored in array1)
While(!Last Image in the database)
    Current Image ID and Image Path take to a 2D array "arrdb"
    Generate histogram to retrieved Image in the database(Histogram values stored in array2)
    Calculate similarity using Intersection Distance measure
    Store the result in dist(distance) array with Image ID
    Retrieve next record from the database
Sort the dist(distance) array elements
Retrieve smallest 9 distances
Retrieve images from the folder corresponding to those distances using Image paths
Display retrieved images

Code. Variable declarations.

```
Image file;
        Bitmap newBitmap, newBitmap2;
        public static int no_images = 127;
        double r, g, b;
        double h, s, v;
        double temp, min, sum;
        int hh, ss, vv;
        int[, ,] count = new int[10, 4, 4];
        //      int[, ,] count1 = new int[10,4,4] ;
        //      int[, ,] count2 = new int[10, 4, 4];
        double[] array1 = new double[10 * 4 * 4];
        double[] array2 = new double[10 * 4 * 4];
        string[] array3 = new string[10 * 4 * 4];
        double[] array4 = new double[10 * 4 * 4];
        //   double [,] mtx;
        //    int arrlen;
        double distance;
        double[,] dist = new double[2, no_images];
        double[] search = new double[no_images];
        double[] distSort = new double[no_images];
        //int[] numbers = new int[5] { 1, 5, 2, 4, 3 };
        int[] arrid = new int[9];
        String[,] arrdb = new String[2, no_images];
        String[] paths = new String[9];
        String[] histDb = new String[no_images];
        String string_hist;
```

Retrieving Images using Histogram Intersection Distance.

```
        private void button4_Click(object sender, EventArgs e)
        {
            String connstring = "server=localhost;database=dbcbir2;uid=root";
            MySqlConnection conn = new MySqlConnection(connstring);
            MySqlCommand command = conn.CreateCommand();

            command.CommandText = "select * from images;";

            try
            {
                conn.Open();
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex.Message);
            }

            MySqlDataReader reader = command.ExecuteReader();


            int cnt = 0;

            Form3 c = new Form3();
            count = c.pro(newBitmap);
            array1 = c.hist(count);
            for (int i = 0; i < 10; i++)
                for (int j = 0; j < 4; j++)
                    for (int k = 0; k < 4; k++)
                    {
                        count[i, j, k] = 0;
                    }

            int num = 0;
            while (reader.Read())
            {
                arrdb[0, cnt] = reader["imgid"].ToString();
                arrdb[1, cnt] = reader["imgpath"].ToString();
                if (cnt == num)
                {
                    file = Image.FromFile(reader["imgpath"].ToString());
                    newBitmap2 = new Bitmap(reader["imgpath"].ToString());

                    count = c.pro(newBitmap2);
                    array2 = c.hist(count);
                    for (int i = 0; i < 10; i++)
                        for (int j = 0; j < 4; j++)
                            for (int k = 0; k < 4; k++)
                            {
                                count[i, j, k] = 0;
                            }

                    Intr id = new Intr();
                    double distance = id.intrdist(array1, array2);
```

```
                dist[0, cnt] = cnt + 1;
                dist[1, cnt] = Math.Round(distance, 6);
                dist[1, cnt] = Math.Abs(dist[1, cnt]);
                distSort[cnt] = Math.Round(distance, 6);
                distSort[cnt] = Math.Abs(distSort[cnt]);
                search[cnt] = Math.Round(distance, 6);
                search[cnt] = Math.Abs(search[cnt]);

            }
            cnt++;
            num++;
        }

        Array.Sort(distSort);

        int index = distSort.Length;

        for (int k = 0; k < 9; k++)
        {
            for (int i = 0; i < index; i++)
            {
                if (dist[1, i] == distSort[k])
                {
                    arrid[k] = (int)dist[0, i];
                    break;
                }
            }
        }

        conn.Close();

        for (int k = 0; k < 9; k++)
        {
            for (int i = 0; i < 127; i++)
            {
                if (arrdb[0, i] == arrid[k].ToString())
                {
                    paths[k] = arrdb[1, i];
                    break;
                }
            }
        }

        pictureBox2.Image = Image.FromFile(paths[0]);
        pictureBox3.Image = Image.FromFile(paths[1]);
        pictureBox4.Image = Image.FromFile(paths[2]);
        pictureBox5.Image = Image.FromFile(paths[3]);
        pictureBox6.Image = Image.FromFile(paths[4]);
        pictureBox7.Image = Image.FromFile(paths[5]);
        pictureBox8.Image = Image.FromFile(paths[6]);
        pictureBox9.Image = Image.FromFile(paths[7]);
        pictureBox10.Image = Image.FromFile(paths[8]);
    }
```

Drawback of the Current System

It takes a significant amount of time to retrieve images with the existing system. The current system takes an image from the user and compares it with every image in the database. This take a longer time since it is a long process and databases usually have a number of images. The process includes, changing the color model from RGB to HSV, calculating and normalizing the histogram, quantizing each pixel into bins, calculating the similarity metric between query image and images from database and retrieving similar images.

Performance Improvement for the Current System

In the proposed system, an intermediate result of each image is stored in the database instead of full images. For each image, quantized bin values are stored in the database. It will certainly speed up the process since when the system needs to retrieve images, more than half of the work is already complete and it just needs to take the generated values and compare with the user inserted image.

Proposed method to improve the performance of the system, is shown in Figure 8.

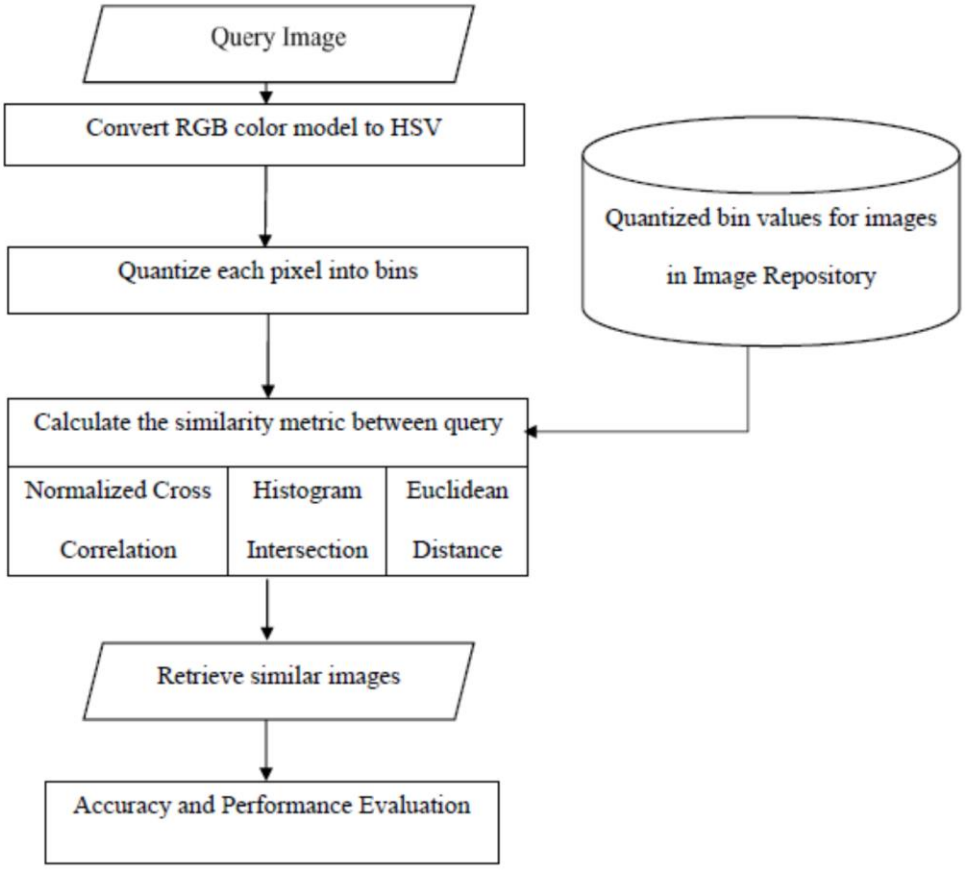

Figure 8. Proposed methodology to retrieve images.

Implementation of the Proposed System

Database. Figure 9 shows the modified database structure of the proposed content based image retrieval system.

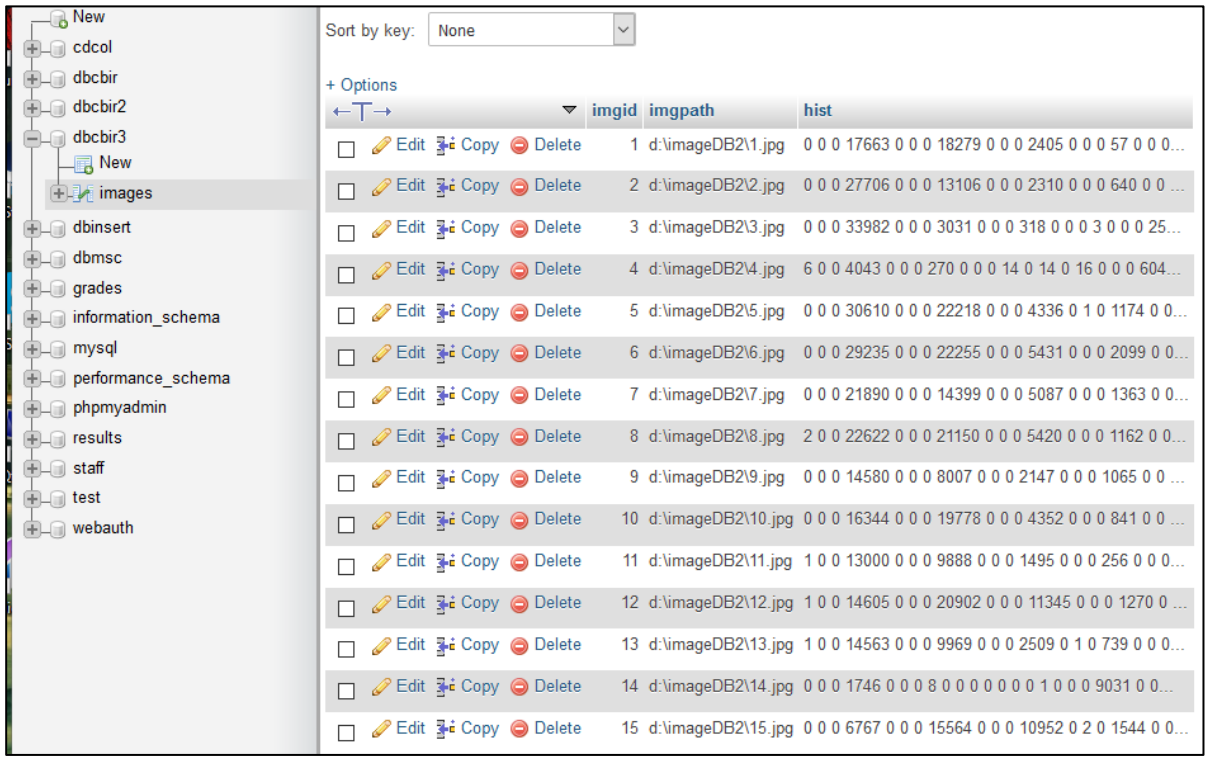

Figure 9. Database structure for proposed system

Update database–algorithm. In this system, histograms were stored using an event. It is possible to modify this method to add any number of images and store their histogram just by clicking on a button.

```
Connect to Database
While(!Last Image in the database)
        Generate histogram to retrieved Image in the database(Histogram values stored
    in array2)
        Copy all the values to histDb[] string array
Close the Connection
Connect to Database again
         Update the database with histDb[] values
 Close the Connection
```

Update database–code.

```
private void button5_Click(object sender, EventArgs e)
        {
            String connstring = "server=localhost;database=dbcbir3;uid=root;";
            MySqlConnection conn = new MySqlConnection(connstring);
            MySqlCommand command = conn.CreateCommand();

            command.CommandText = "select * from images;";

            try
            {
                conn.Open();
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex.Message);
            }

            MySqlDataReader reader = command.ExecuteReader();


            int cnt = 0;

            Form3 c = new Form3();
            count = c.pro(newBitmap);
            array1 = c.hist(count);
            for (int i = 0; i < 10; i++)
                for (int j = 0; j < 4; j++)
                    for (int k = 0; k < 4; k++)
                    {
                        count[i, j, k] = 0;
                    }

            int num = 0;
            int m = 0;


            while (reader.Read())
            {
                arrdb[0, cnt] = reader["imgid"].ToString();
                arrdb[1, cnt] = reader["imgpath"].ToString();
                if (cnt == num)
                {

                    file = Image.FromFile(reader["imgpath"].ToString());
                    newBitmap2 = new Bitmap(reader["imgpath"].ToString());

                    count = c.pro(newBitmap2);
                    array2 = c.hist(count);

                    int l = array1.Length;
                    String z = l.ToString();

                    string output = " ";
```

```
                    for (int k = 0; k < 160; k++)
                    {
                        output += array2[k].ToString() + " ";
                    }

                     histDb[m] = output;

                    m++;

                    for (int i = 0; i < 10; i++)
                        for (int j = 0; j < 4; j++)
                            for (int k = 0; k < 4; k++)
                            {
                                count[i, j, k] = 0;
                            }

                }
                cnt++;
                num++;

            }


            conn.Close();

            try
            {
                conn.Open();
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex.Message);
            }

            for (int q = 0; q < no_images; q++)
            {
                command.CommandText = "update images set hist='" + histDb[q] + "'
where imgid=" + (q + 1) + ";";
                command.ExecuteNonQuery();
            }

            for(int q = 0; q < no_images+1; q++)
            {
                command.CommandText = "update images set
imgpath='d:\\\\imageDB2\\\\" + q + ".jpg' where imgid=" + q + ";";
                command.ExecuteNonQuery();
            }

            conn.Close();

        }
```

In this system, histograms were stored using an event. It is possible to modify this method to add any number of images and store their histogram just by clicking on a button.

Making changes to retrieval method.

Improved method–algorithm.

Connect to Database
Get all the records in Image table to Reader object
Generate histogram to user entered Image(Histogram values stored in array1)
While(!Last Image in the database
    //Current Image ID and Image Path take to a 2D array "arrdb"
    //Generate histogram to retrieved Image in the database(Histogram values stored in array2)
    Take the histogram values from the database to a string
    Spilt the string and place numbers in an array

    Calculate similarity using Intersection Distance measure
    Store the result in dist(distance) array with Image ID
    Retrieve next record from the database
Sort the dist(distance) array elements
Retrieve smallest 9 distances

Improved method for Histogram Intersection Distance–code.

```
private void button7_Click(object sender, EventArgs e)
        {

            //Implementing a Stop Watch to Track Time
            Stopwatch stopWatch = new Stopwatch();
            stopWatch.Start();

            String connstring = "server=localhost;database=dbcbir3;uid=root";
            MySqlConnection conn = new MySqlConnection(connstring);
            MySqlCommand command = conn.CreateCommand();

            command.CommandText = "select * from images;";

            try
            {
                conn.Open();
            }
            catch (Exception ex)
            {
```

```
                Console.WriteLine(ex.Message);
            }

            MySqlDataReader reader = command.ExecuteReader();


            int cnt = 0;

            Form3 c = new Form3();
            count = c.pro(newBitmap);
            array1 = c.hist(count);
            for (int i = 0; i < 10; i++)
                for (int j = 0; j < 4; j++)
                    for (int k = 0; k < 4; k++)
                    {
                        count[i, j, k] = 0;
                    }

            int num = 0;
            while (reader.Read())
            {
                arrdb[0, cnt] = reader["imgid"].ToString();
                arrdb[1, cnt] = reader["imgpath"].ToString();
                if (cnt == num)
                {

                    file = Image.FromFile(reader["imgpath"].ToString());

                    //Take the histogram values from the database
                    string_hist = reader["hist"].ToString();

                    //Split numbers
                    array3 = string_hist.Split(new string[] { " " },
StringSplitOptions.None);

                    for (int k = 0; k < 160; k++)
                    {
                        //Convert string to int
                        array4[k] = Double.Parse(array3[k + 1]);
                    }

                    for (int i = 0; i < 10; i++)
                        for (int j = 0; j < 4; j++)
                            for (int k = 0; k < 4; k++)
                            {
                                count[i, j, k] = 0;
                            }

                    Intr id = new Intr();
                    double distance=id.intrdist(array1,array4);
                    dist[0, cnt] = cnt + 1;
                    dist[1, cnt] = Math.Round(distance, 6);
                    dist[1, cnt] = Math.Abs(dist[1, cnt]);
                    distSort[cnt] = Math.Round(distance, 6);
                    distSort[cnt] = Math.Abs(distSort[cnt]);
```

```
                search[cnt] = Math.Round(distance, 6);
                search[cnt] = Math.Abs(search[cnt]);
            }
            cnt++;
            num++;
        }

        Array.Sort(distSort);



        int index = distSort.Length;

        for (int k = 0; k < 9; k++)
        {
            for (int i = 0; i < index; i++)
            {
                if (dist[1, i] == distSort[k])
                {
                    arrid[k] = (int)dist[0, i];
                    break;
                }
            }
        }

        conn.Close();

        for (int k = 0; k < 9; k++)
        {
            for (int i = 0; i < no_images; i++)
            {
                if (arrdb[0, i] == arrid[k].ToString())
                {
                    paths[k] = arrdb[1, i];
                    break;
                }
            }
        }

        pictureBox2.Image = Image.FromFile(paths[0]);
        pictureBox3.Image = Image.FromFile(paths[1]);
        pictureBox4.Image = Image.FromFile(paths[2]);
        pictureBox5.Image = Image.FromFile(paths[3]);
        pictureBox6.Image = Image.FromFile(paths[4]);
        pictureBox7.Image = Image.FromFile(paths[5]);
        pictureBox8.Image = Image.FromFile(paths[6]);
        pictureBox9.Image = Image.FromFile(paths[7]);
        pictureBox10.Image = Image.FromFile(paths[8]);


        stopWatch.Stop();

        //Get the elapsed time as a TimeSpan value
        TimeSpan ts = stopWatch.Elapsed;
```

```
            //Format and display the TimeSpan value.
            String elapsedTime = String.Format("{0:00}:{1:00}:{2:00}.{3:00}",
            ts.Hours, ts.Minutes, ts.Seconds,
            ts.Milliseconds / 10);
            MessageBox.Show("Time Taken\nHrs:Mins:Secs:MilSecs\n" + elapsedTime);
    }
```

Improved method for Normalized Cross-Correlation–code

```
private void button6_Click(object sender, EventArgs e)
        {
            //Implementing a Stop Watch to Track Time
            Stopwatch stopWatch = new Stopwatch();
            stopWatch.Start();

            String connstring = "server=localhost;database=dbcbir3;uid=root";
            MySqlConnection conn = new MySqlConnection(connstring);
            MySqlCommand command = conn.CreateCommand();

            command.CommandText = "select * from images;";

            try
            {
                conn.Open();
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex.Message);
            }

            MySqlDataReader reader = command.ExecuteReader();


            int cnt = 0;

            Form3 c = new Form3();
            count = c.pro(newBitmap);
            array1 = c.hist(count);
            for (int i = 0; i < 10; i++)
                for (int j = 0; j < 4; j++)
                    for (int k = 0; k < 4; k++)
                    {
                        count[i, j, k] = 0;
                    }

            int num = 0;

            while (reader.Read())
            {
                arrdb[0, cnt] = reader["imgid"].ToString();
                arrdb[1, cnt] = reader["imgpath"].ToString();

                if (cnt == num)
                {
```

```
                    file = Image.FromFile(reader["imgpath"].ToString());

                    //Take the histogram values from the database
                    string_hist = reader["hist"].ToString();

                    //Split numbers
                    array3 = string_hist.Split(new string[] { " " },
StringSplitOptions.None);

                    for (int k = 0; k < 160; k++)
                    {
                        //Convert string to int
                        array4[k] = Double.Parse(array3[k+1]);
                    }

                    int l = array1.Length;
                    String z = l.ToString();

                    string output = " ";


                    for (int k = 0; k < 160; k++)
                    {
                        output += array1[k].ToString() + " ";
                    }

                    for (int i = 0; i < 10; i++)
                        for (int j = 0; j < 4; j++)
                            for (int k = 0; k < 4; k++)
                            {
                                count[i, j, k] = 0;
                            }

                    dist d = new dist(array1);
                    distance = d.cdist(array4);
                    dist[0, cnt] = cnt + 1;
                    dist[1, cnt] = Math.Round(distance, 6);
                    dist[1, cnt] = Math.Abs(dist[1, cnt]);
                    distSort[cnt] = Math.Round(distance, 6);
                    distSort[cnt] = Math.Abs(distSort[cnt]);
                    search[cnt] = Math.Round(distance, 6);
                    search[cnt] = Math.Abs(search[cnt]);

                }
                cnt++;
                num++;
            }

            Array.Sort(distSort);

            int index = distSort.Length;

            for (int k = 0; k < 9; k++)
            {
```

```
                for (int i = 0; i < index; i++)
                {
                    if (dist[1, i] == distSort[k])
                    {
                        arrid[k] = (int)dist[0, i];
                        break;
                    }
                }
            }

            conn.Close();

            for (int k = 0; k < 9; k++)
            {
                for (int i = 0; i < no_images; i++)
                {
                    if (arrdb[0, i] == arrid[k].ToString())
                    {
                        paths[k] = arrdb[1, i];
                        break;
                    }
                }
            }

            pictureBox2.Image = Image.FromFile(paths[0]);
            pictureBox3.Image = Image.FromFile(paths[1]);
            pictureBox4.Image = Image.FromFile(paths[2]);
            pictureBox5.Image = Image.FromFile(paths[3]);
            pictureBox6.Image = Image.FromFile(paths[4]);
            pictureBox7.Image = Image.FromFile(paths[5]);
            pictureBox8.Image = Image.FromFile(paths[6]);
            pictureBox9.Image = Image.FromFile(paths[7]);
            pictureBox10.Image = Image.FromFile(paths[8]);

            stopWatch.Stop();

            //Get the elapsed time as a TimeSpan value
            TimeSpan ts = stopWatch.Elapsed;

            //Format and display the TimeSpan value.
            String elapsedTime = String.Format("{0:00}:{1:00}:{2:00}.{3:00}",
            ts.Hours, ts.Minutes, ts.Seconds,
            ts.Milliseconds / 10);
            MessageBox.Show("Time Taken\nHrs:Mins:Secs:MilSecs\n" + elapsedTime);
        }
```

Image retrieval–database with 1000 images. To retrieve images from the large database has to do a small modification to the program. But using Update event first of all bin values should store in a database. It will take some to store those data. But retrieval process is really fast. The following code shows the variable modified variable declarations and assignments for the program.

```
Image file;
        Bitmap newBitmap, newBitmap2;
        public static int no_images = 999;
        double r, g, b;
        double h, s, v;
        double temp, min, sum;
        int hh, ss, vv;
        int[, ,] count = new int[10, 4, 4];
        //      int[, ,] count1 = new int[10,4,4] ;
        //      int[, ,] count2 = new int[10, 4, 4];
        double[] array1 = new double[10 * 4 * 4];
        double[] array2 = new double[10 * 4 * 4];
        string[] array3 = new string[10 * 4 * 4];
        double[] array4 = new double[10 * 4 * 4];
        //   double [,] mtx;
        //    int arrlen;
        double distance;
        double[,] dist = new double[2, no_images];
        double[] search = new double[no_images];
        double[] distSort = new double[no_images];
        //int[] numbers = new int[5] { 1, 5, 2, 4, 3 };
        int[] arrid = new int[9];
        String[,] arrdb = new String[2, no_images];
        String[] paths = new String[9];
        String[] histDb = new String[no_images];
        String string_hist;
```

Chapter 4: EXPERIMENTAL RESULTS AND DISCUSSION

In this section experimental results for each distance measure and performance improvement with the proposed image retrieval technique discussed. To compare distances an image inserted as the query image and distances were taken for each image in the database. Figure 10 shows the Graphical User Interface of the system to compare distance measures. Pop-up window shows the distance values between the query image and the images in the database.

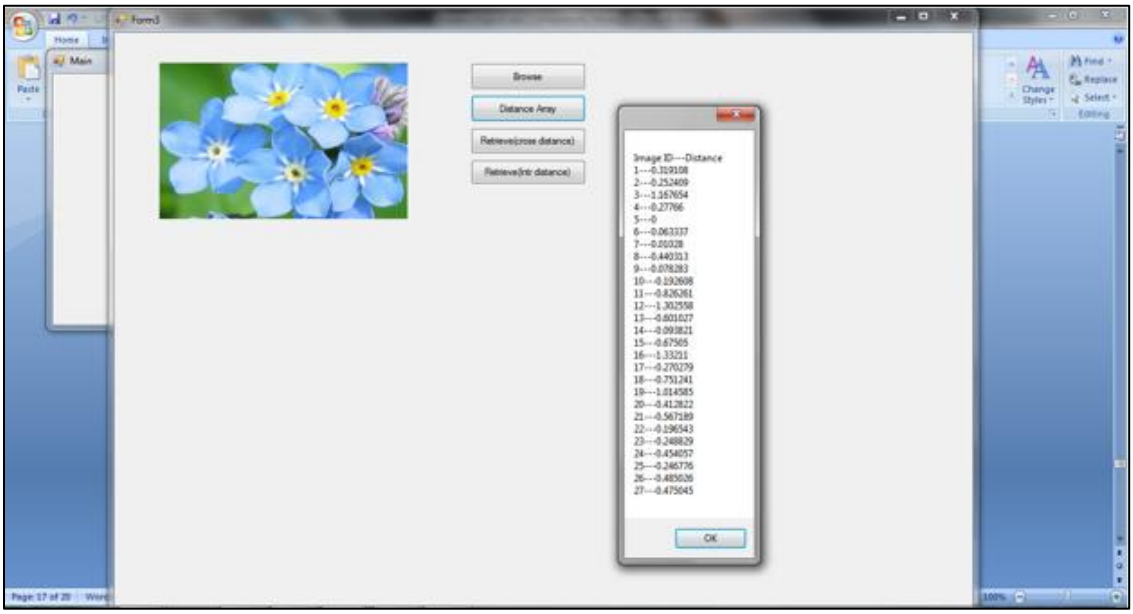

Figure 10. GUI of the system created to get the distance measures to company images.

Analyzing Distance Measures

Distance values for each image were entered to MINITAB worksheet and plot. The distributions of the distance values were then analyzed. Similar images are shown in Red color on the plot.

Sample Image 1.

Database and similar images.

A blue color flower image (5.jpg) is taken as the query image to compare distances with other images. Figure 11 shows the similar images for query image (5.jpg) in the database.

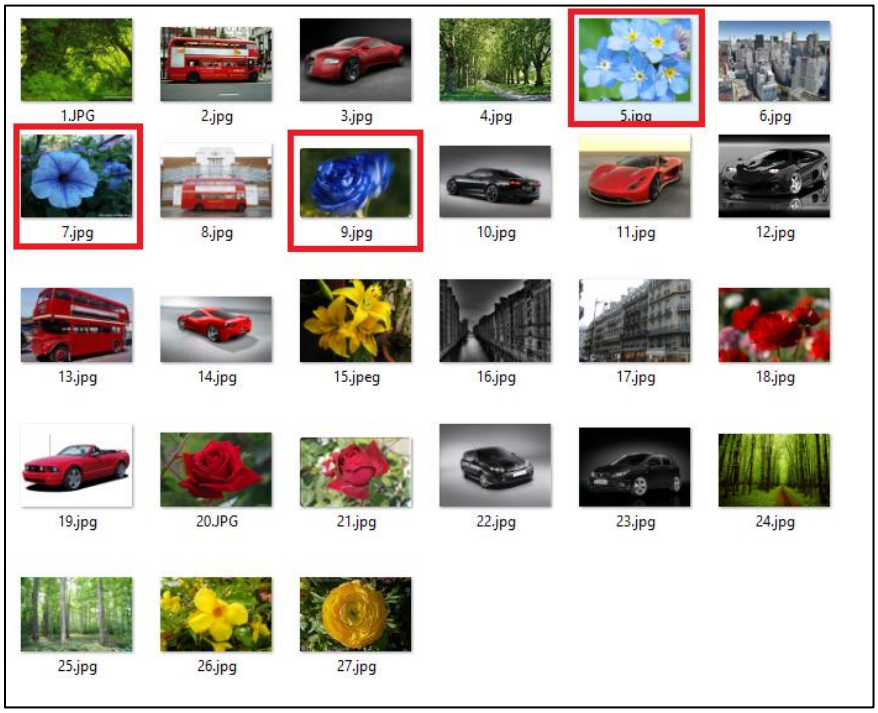

Figure 11. Image database to analyze distance measure–sample image 1.

Figure 12, 13 and 14 shows the different distance measures distribution for a sample image 1. Similar images are shown in red color dots and other images are shown in black color dots in the plot. In the database, the query image was included to make sure the program is working appropriately. For that image distance, measure should be 0. This analysis helped to differentiate behaviors in each distance measure.

Euclidean distance.

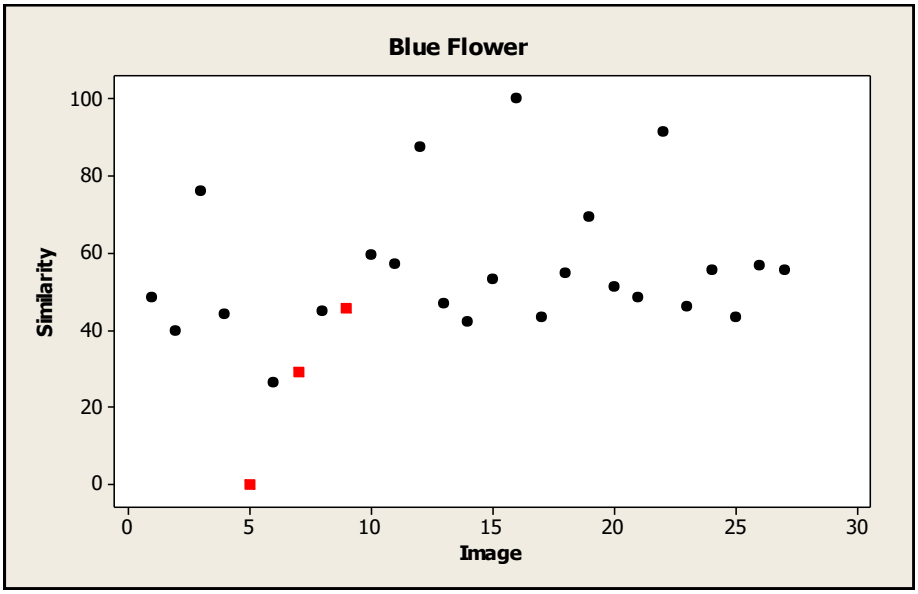

Figure 12. Comparing distances using Euclidean distance for sample image 1.

Histogram intersection distance.

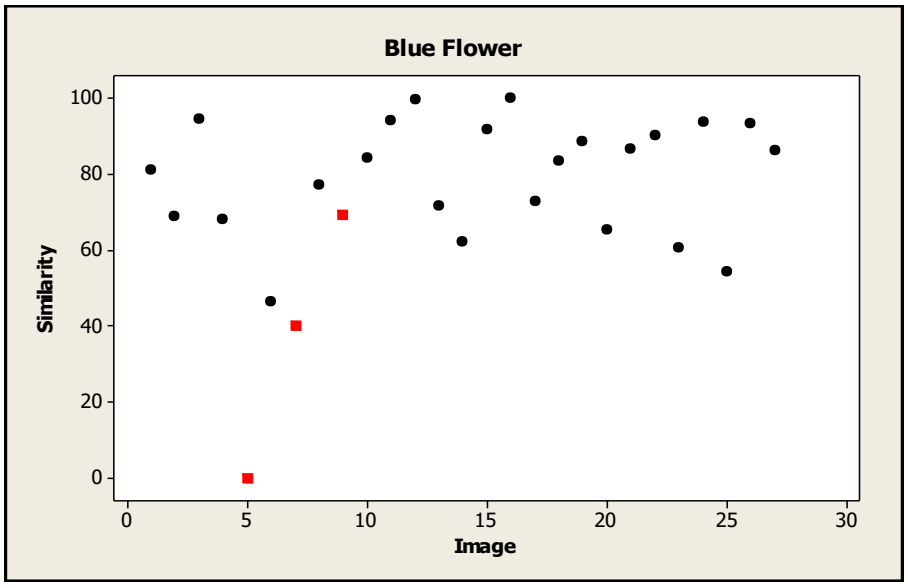

Figure 13. Comparing distances using Histogram Intersection distance for sample image 1.

Normalized cross correlation distance.

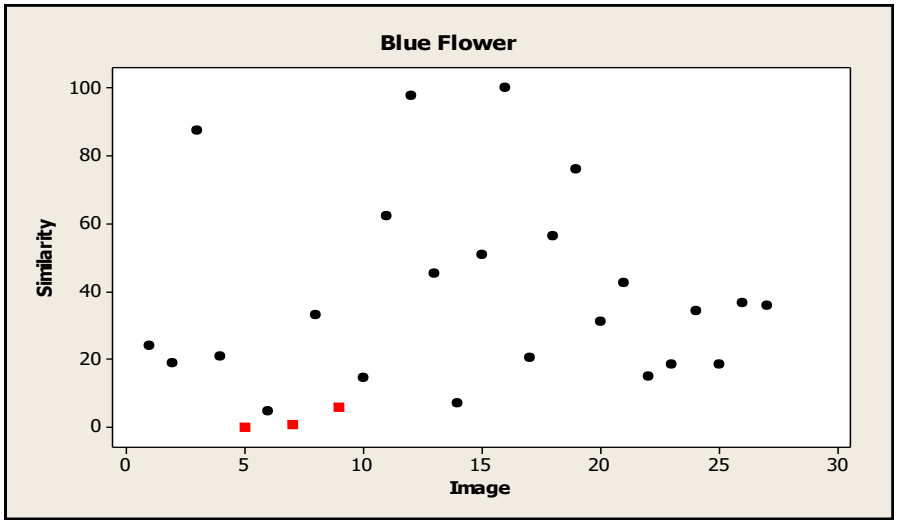

Figure 14. Comparing distances using Cross-Correlation distance for sample image 1.

Sample Image 2.

Database and similar images.

An image with trees (1.jpg) is taken as the query image to compare distances with other images. Figure 15 shows the similar images for query image (1.jpg) in the database.

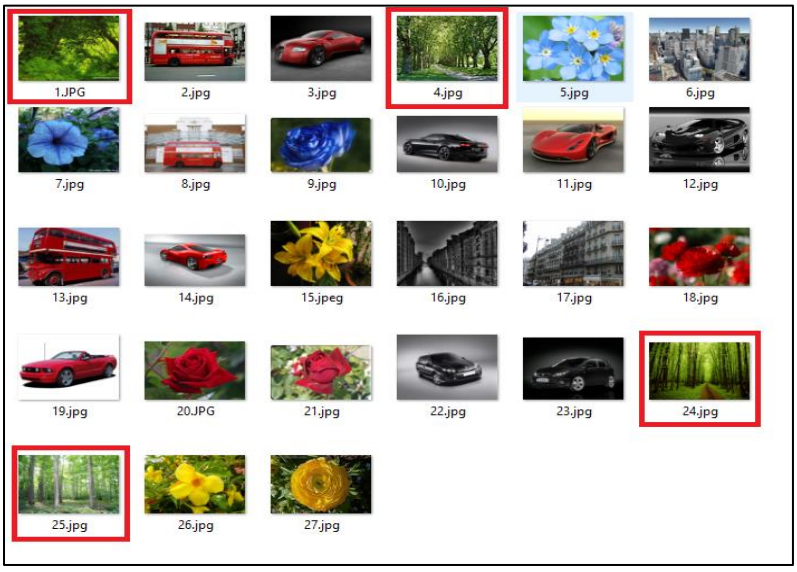

Figure 15. Image database to analyze distance measure–sample image 2.

Figure 16, 17, and 18 shows the different distance measures distribution for a sample image 1. Similar images are shown in red color dots and other images are shown in black color dots in the plot.

Euclidean distance.

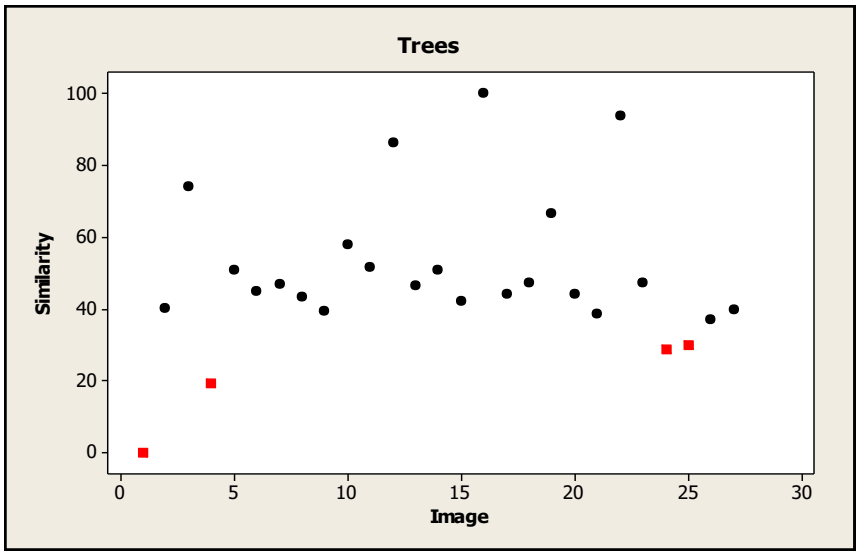

Figure 16. Comparing distances using Euclidean distance for sample image 2.

Histogram intersection distance.

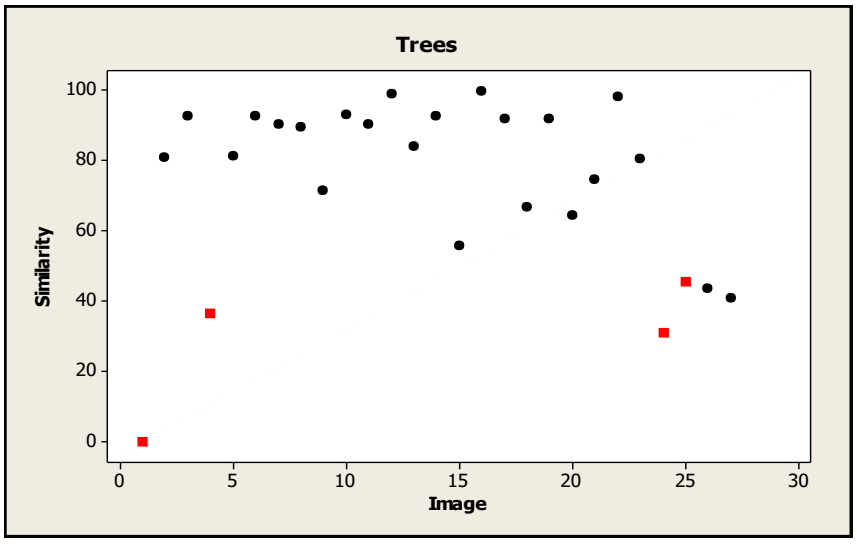

Figure 17. Comparing distances using Histogram Intersection distance for sample image 2.

Normalized cross correlation distance.

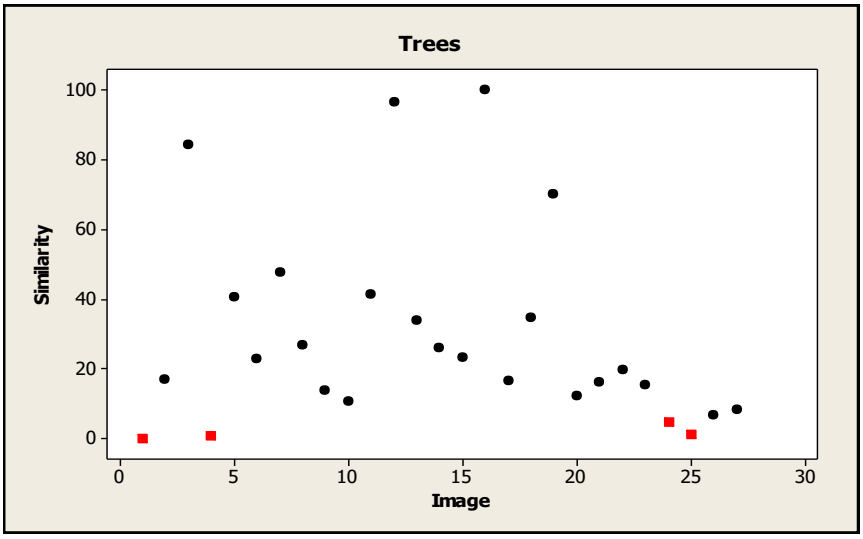

Figure 18. Comparing distances using Cross-Correlation distance for sample image 2.

Image Retrieval–Database with 27 Images

Histogram intersection distance. Figure 19 shows, the image retrieval of the implemented system, using histogram intersection distance. A yellow color flower is taken as the query image.

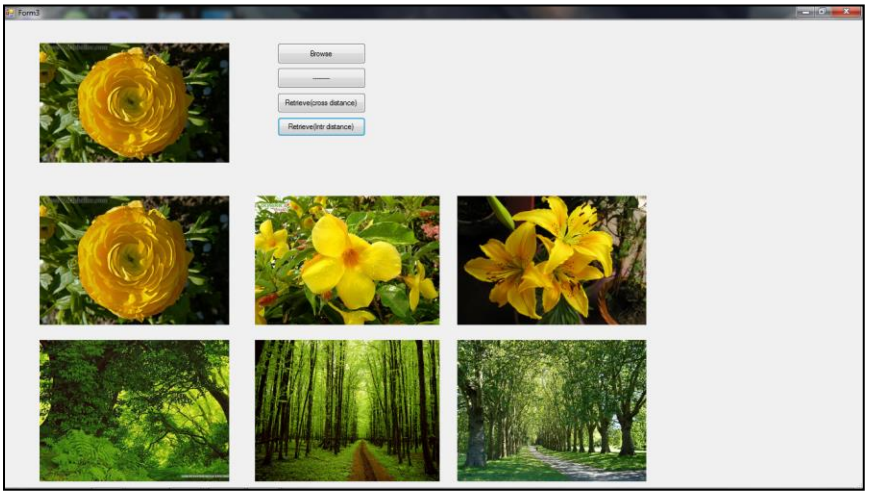

Figure 19. Retrieving images using Histogram Intersection Distance–image database with 27 images.

Normalized cross-correlation distance. Figure 20 shows, the image retrieval of the implemented system, using normalized cross correlation distance. A yellow color flower is taken as the query image.

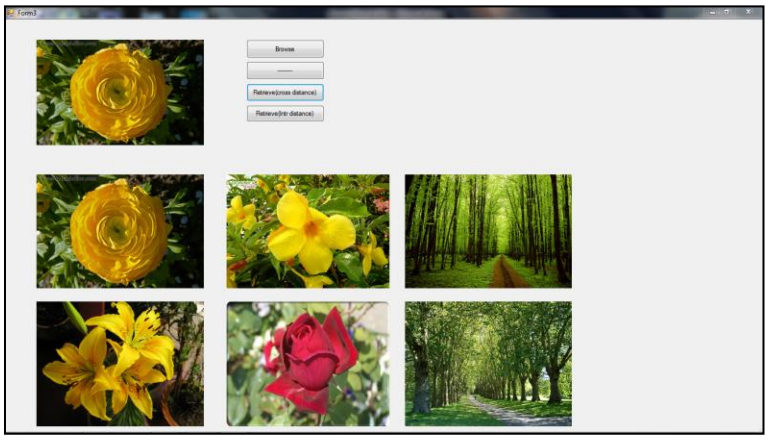

Figure 20. Retrieving images using Cross-Correlation distance–image database with 27 images.

Computer System Environment

The implemented methods were tested in a laptop computer. Some basic information about the system environment is listed below.

Hardware.

Processor: Intel ® Core (TM) i7-4710HQ CPU @ 2.50 GHz

Memory (RAM): 12.0 GB

Hard Disk: HDD

Software.

Operating System: Windows 10

Other software:

Microsoft Visual Studio Community 2015

XAMPP Version 5.6.12

Image Retrieval–Database with 127 Images

To create this database images were taken from an online source. It is specially developed for research comparisons [20] [21] [22]. To find out the processing time, a simple stop watch attached to the code [23].

Histogram intersection distance. Figure 21 shows, the image retrieval of first version of content based image retrieval system, using histogram intersection distance. Processing time is also shown in the figure.

Processing Time = 15.98 Secs

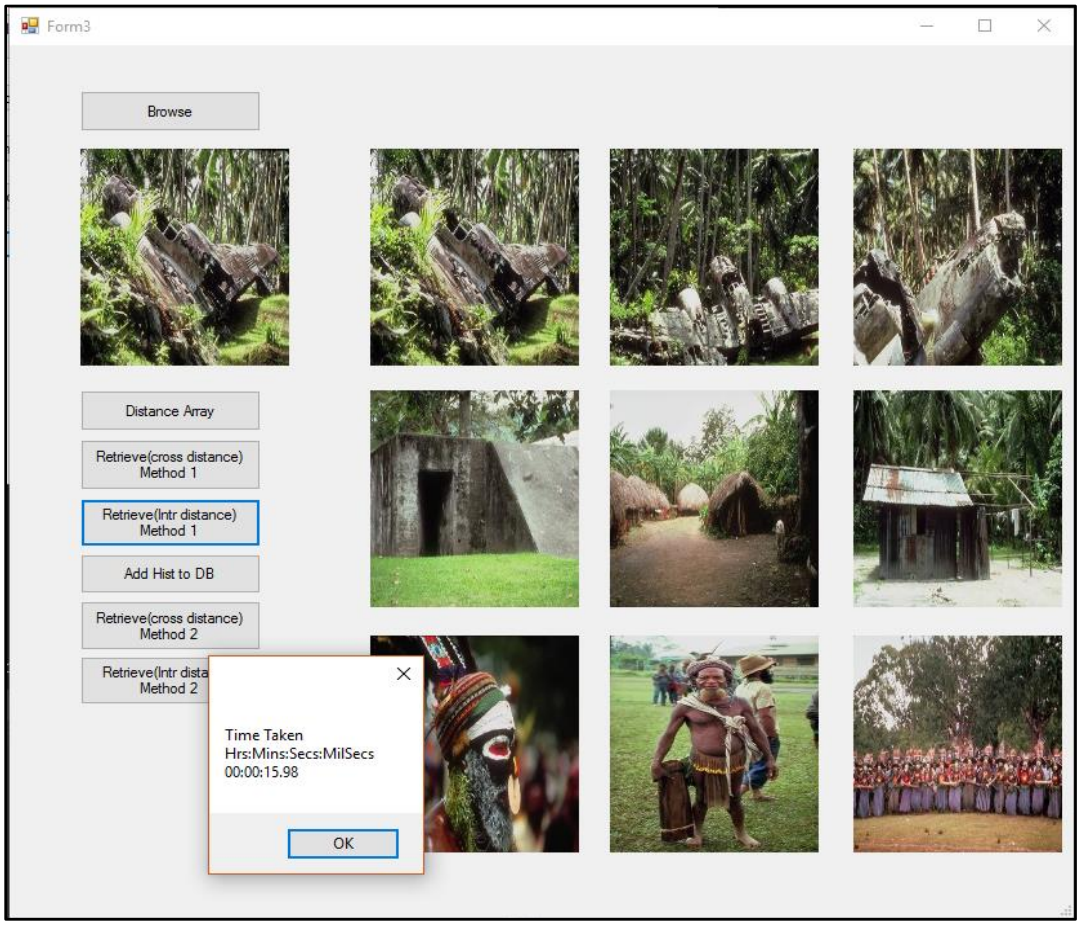

Figure 21. Retrieving images using Histogram Intersection distance–image database with 127 images.

Normalized cross correlation distance. Figure 22 shows, the image retrieval of first version of content based image retrieval system, using normalized cross correlation distance. Processing time is also shown in the figure.

Processing Time = 14.10 Secs

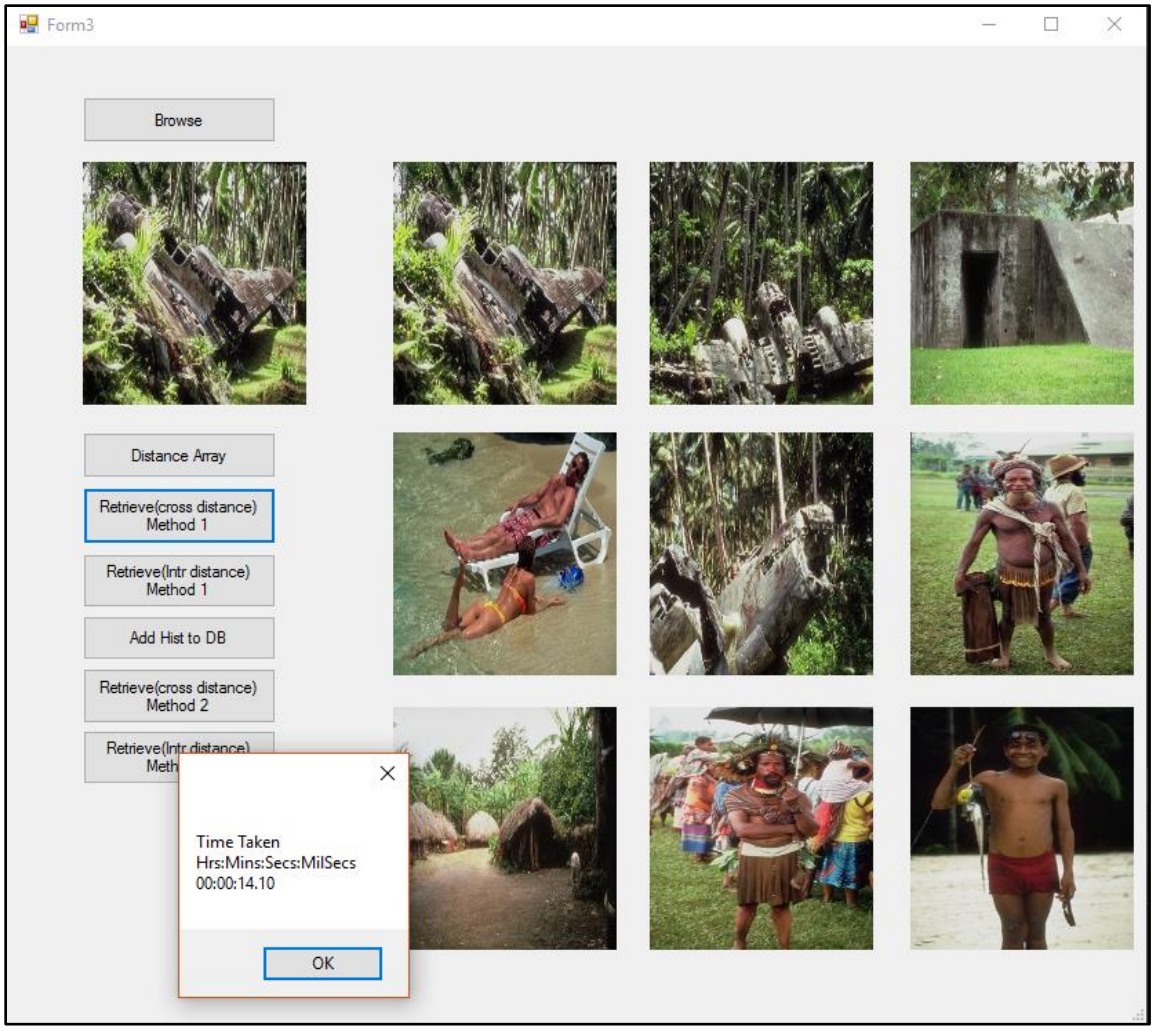

Figure 22. Retrieving images using Cross-Correlation distance–image database with 127 images.

Histogram intersection distance–proposed system. Figure 23 shows, the image retrieval of proposed content based image retrieval system, using histogram intersection distance. Processing time is also shown in the figure.

Processing Time = 00.25 Secs

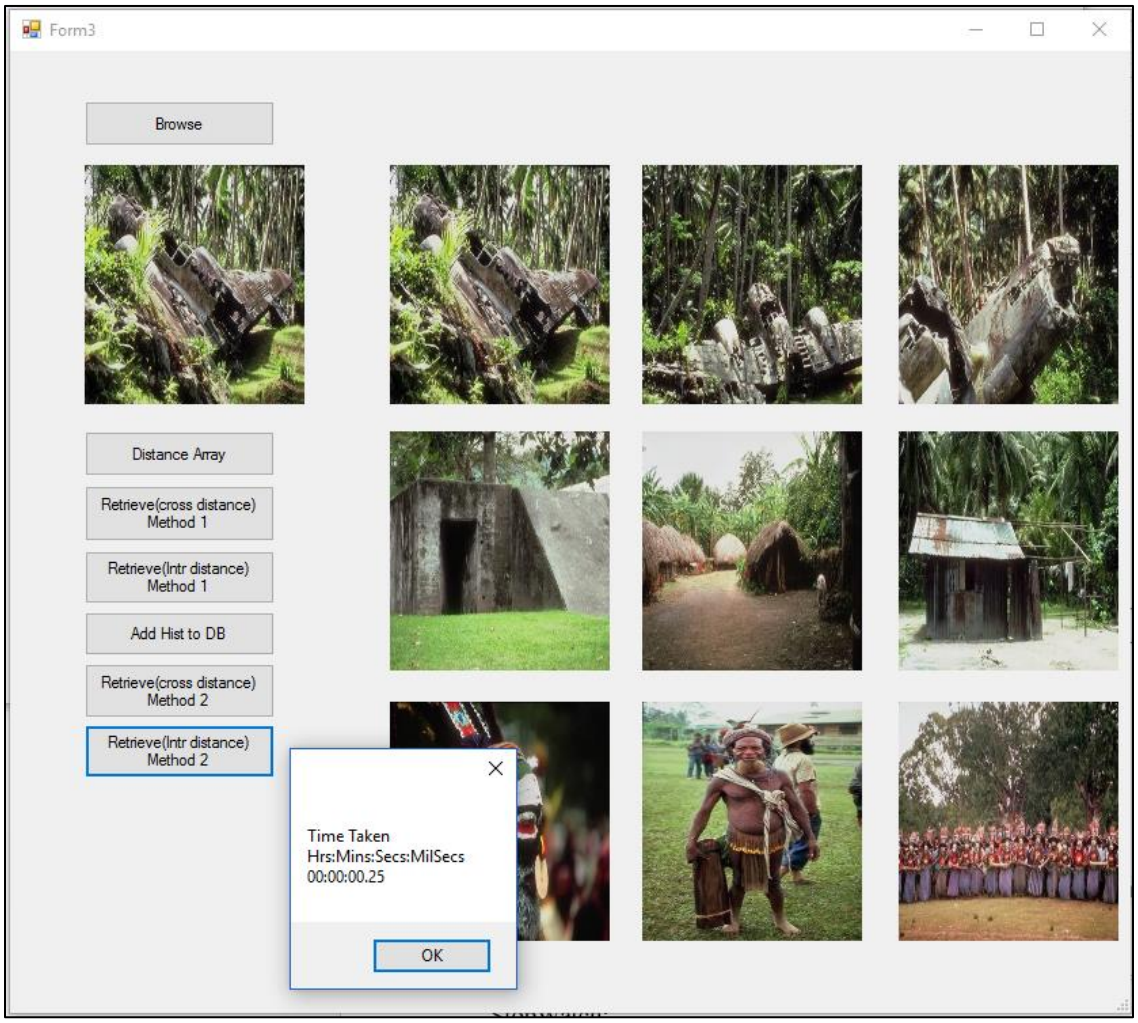

Figure 23. Image retrieval in proposed system–using histogram intersection distance.

Normalized cross correlation distance–proposed system. Figure 24 shows, the image retrieval of proposed content based image retrieval system, using normalized cross correlation distance. Processing time is also shown in the figure.

Processing Time = 00.35 Secs

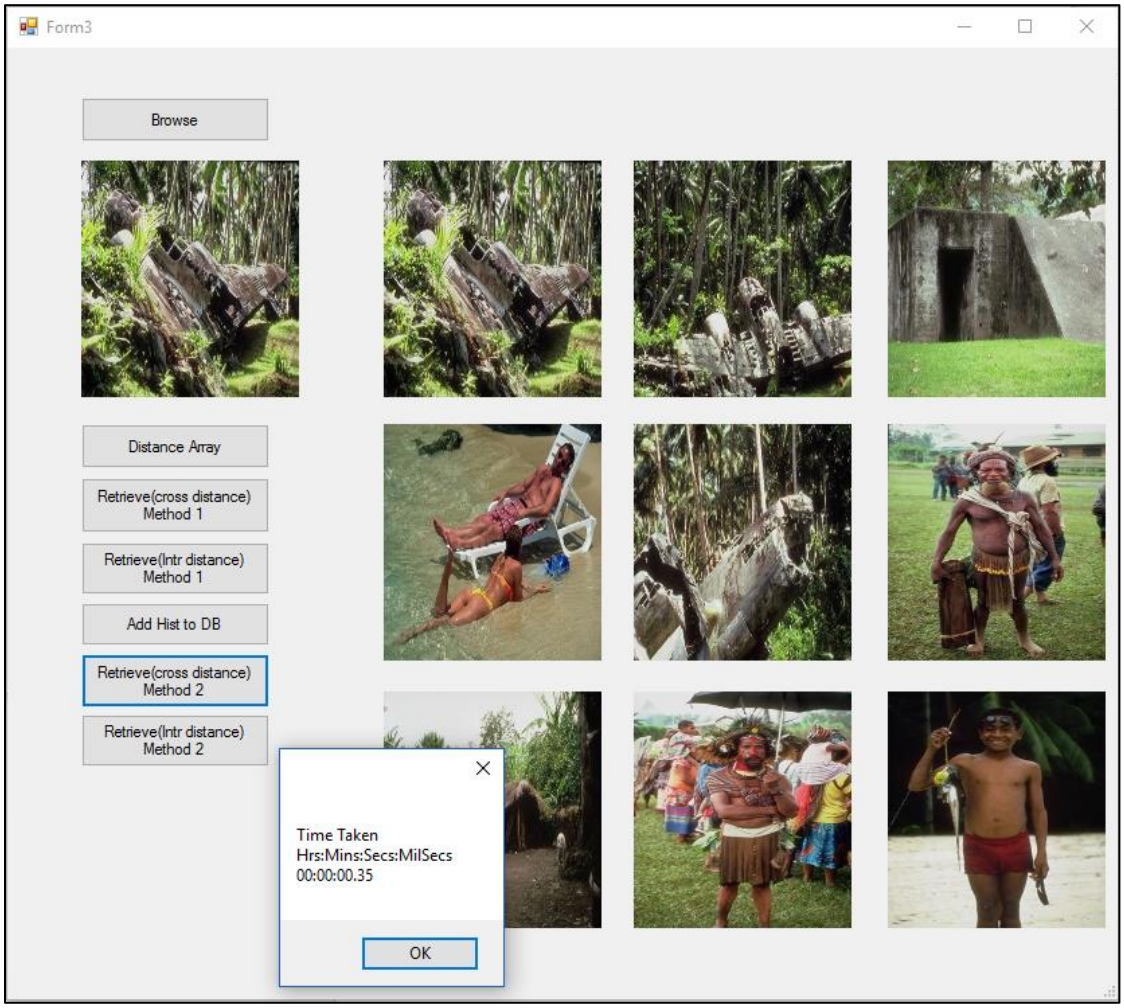

Figure 24. Image retrieval in proposed system–using cross-correlation distance.

The proposed system is significantly faster than the existing method.

Comparing the Proposed Image Retrieval System with the Old Image Retrieval System

Comparison in retrieval time, between proposed image retrieval system, and the previous image retrieval system is shown in Table 2.

Table 2

Comparison in Retrieval Time, between Proposed Image Retrieval System and Previous Image Retrieval System

	Conventional image retrieval system Time (in seconds)	Proposed image retrieval system Time (in seconds)
Histogram Intersection Distance	15.98	00.25
Normalized Cross Correlation Distance	14.10	00.35

Image Retrieval–Database with 1000 Images

Finally, a content based image retrieval system developed with 1000 images. Following illustration (Table 3) shows the outcome for different types of images.

Table 3

Evaluation of the Accuracy and the Performance for the Propsed System

Image	Method	Number of similar images retrieved by the system	Time (in seconds)
Red color bus	Normalized Cross Correlation	6/9	02:09
	Histogram Intersection	9/9	01:16
Dinosaur	Normalized Cross Correlation	9/9	02:08
	Histogram Intersection	9/9	01:11
Yellow color flower	Normalized Cross Correlation	5/9	02:11
	Histogram Intersection	9/9	01:16
Two horses on meadow	Normalized Cross Correlation	9/9	02:08
	Histogram Intersection	9/9	01:10

Red color bus.

Normalized Cross correlation distance. Figure 25 shows the result set of the proposed content based image retrieval system, for a red color bus query image. Normalized cross correlation distance used to retrieve following images.

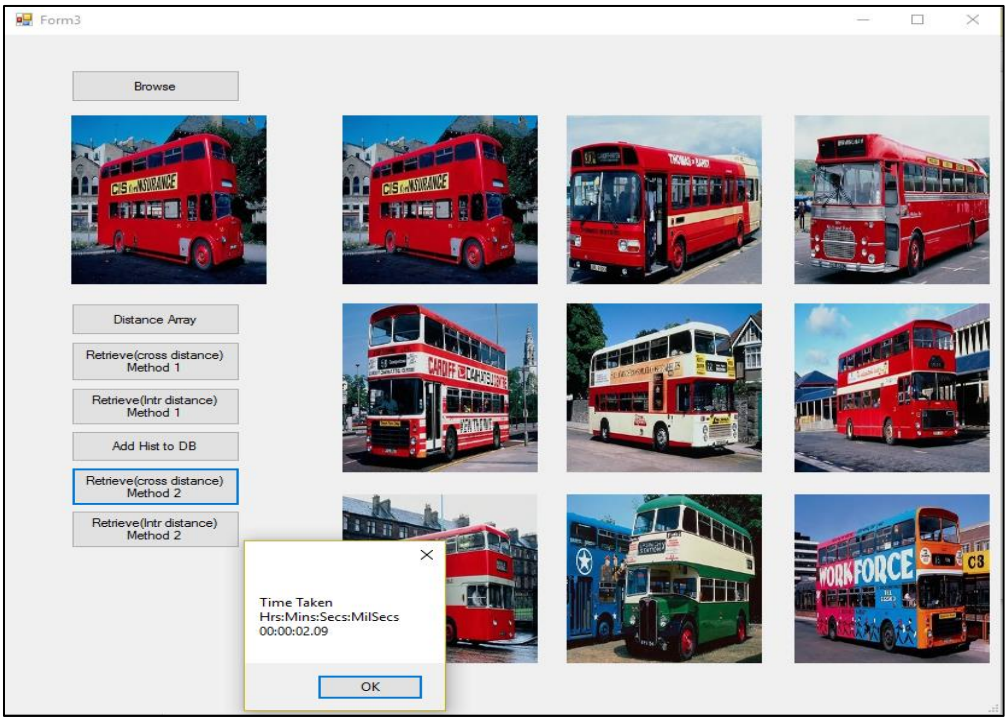

Figure 25. CBIR system–results set 1.

Histogram intersection distance. Figure 26 shows the result set of the proposed content based image retrieval system, for a red color bus query image. Histogram intersection distance used to retrieve following images.

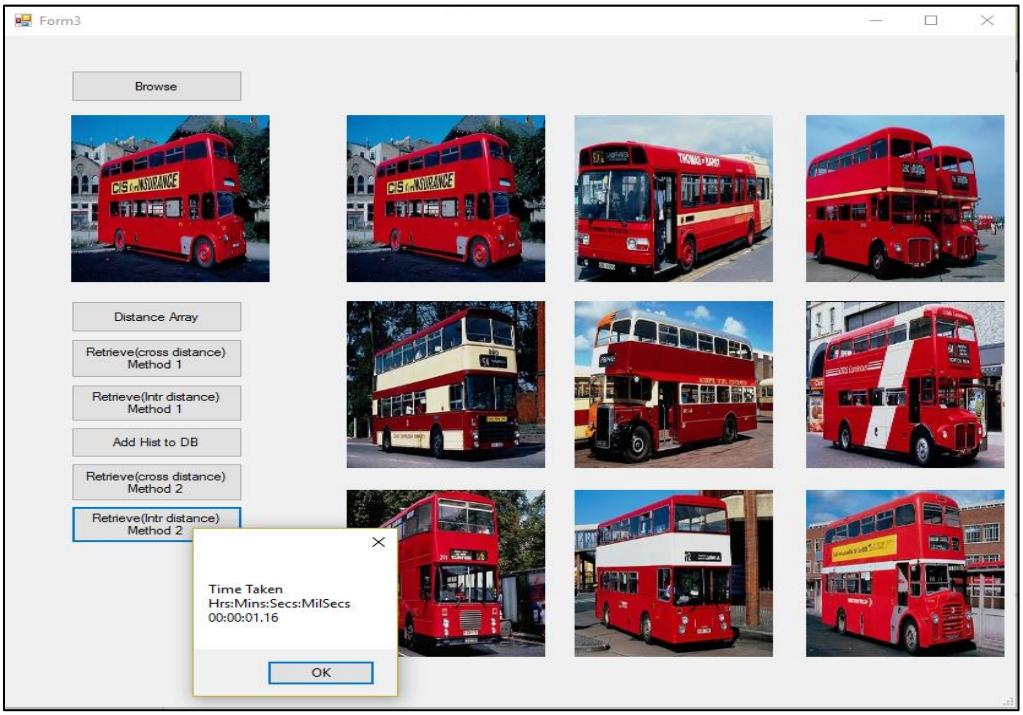

Figure 26. CBIR system–results set 2.

Dinosaur.

Normalized cross correlation. Figure 27 shows the result set of the proposed content based image retrieval system, for a dinosaur query image. Normalized cross correlation distance used to retrieve following images.

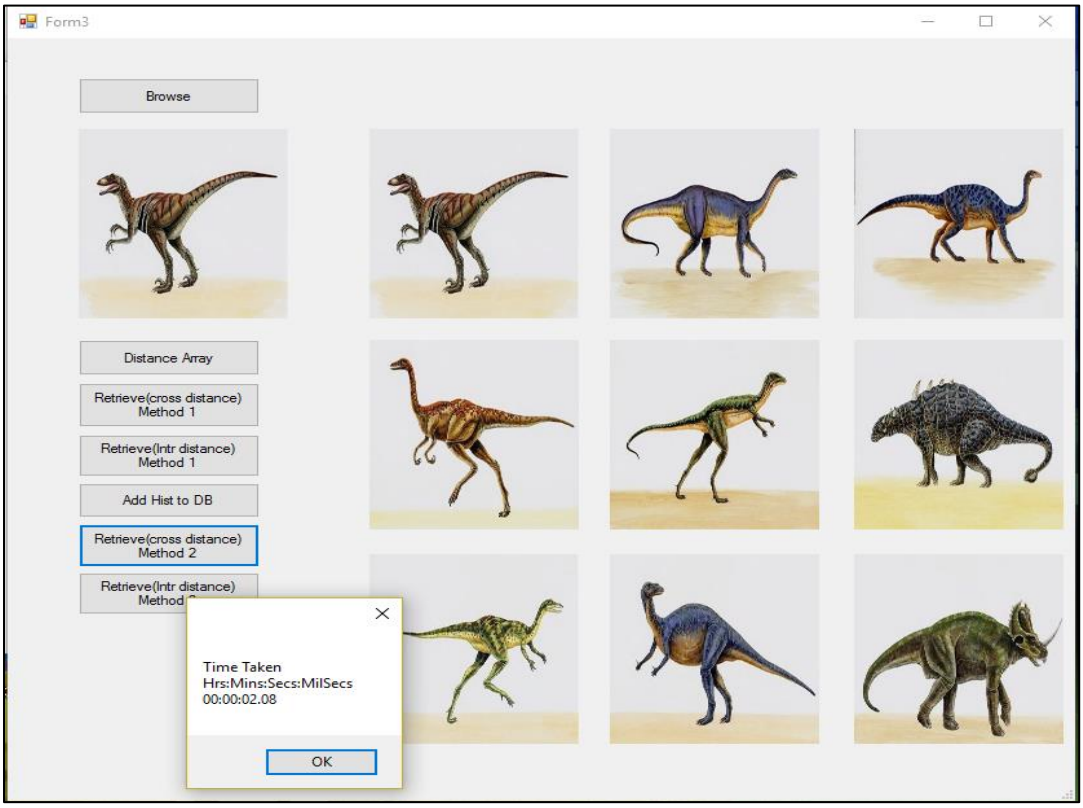

Figure 27. CBIR system–results set 3.

Histogram intersection distance. Figure 28 shows the result set of the proposed content based image retrieval system, for a dinosaur query image. Histogram intersection distance used to retrieve following images.

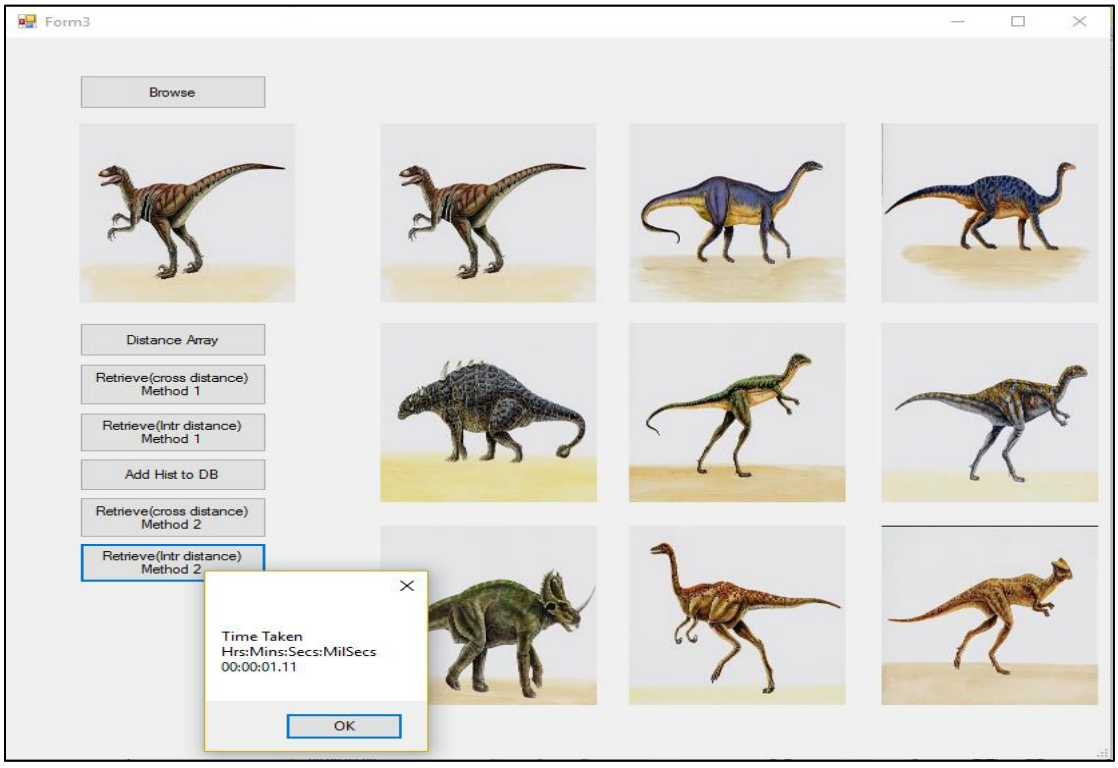

Figure 28. CBIR system–results set 4.

Yellow color flower.

Normalized Cross correlation distance. Figure 29 shows the result set of the proposed content based image retrieval system, for a yellow color flower query image. Normalized cross correlation distance used to retrieve following images.

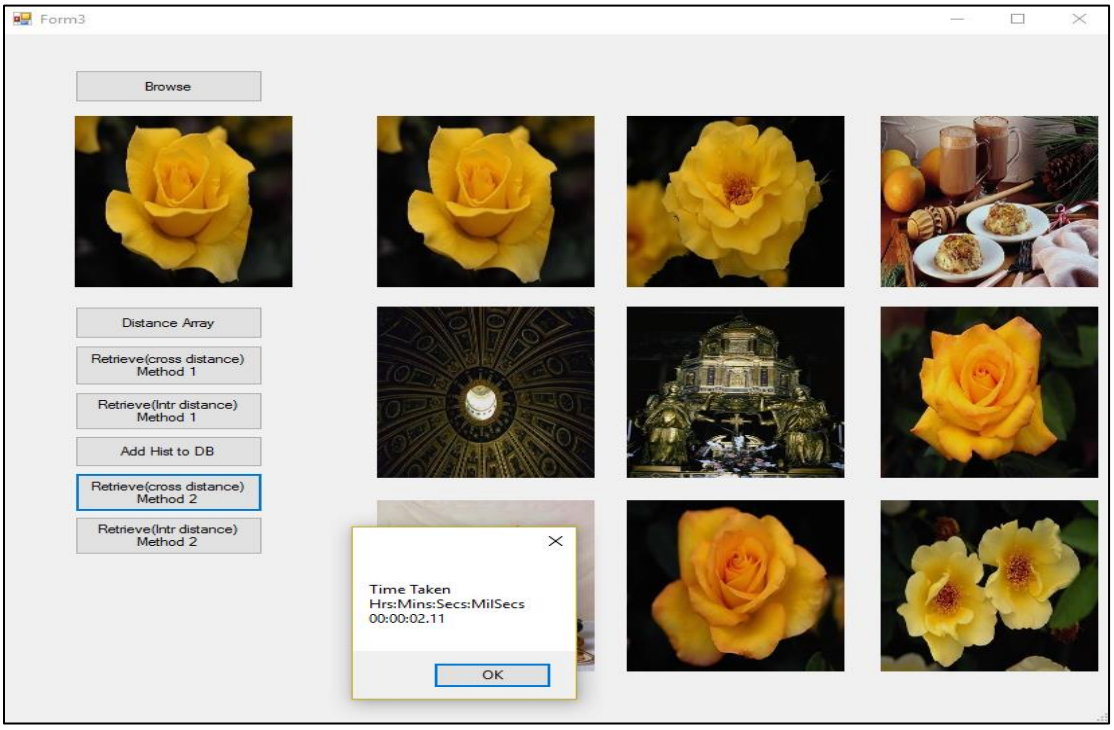

Figure 29. CBIR system–results set 5.

Histogram intersection distance. Figure 30 shows the result set of the proposed content based image retrieval system, for a yellow color flower query image. Histogram intersection distance used to retrieve following images.

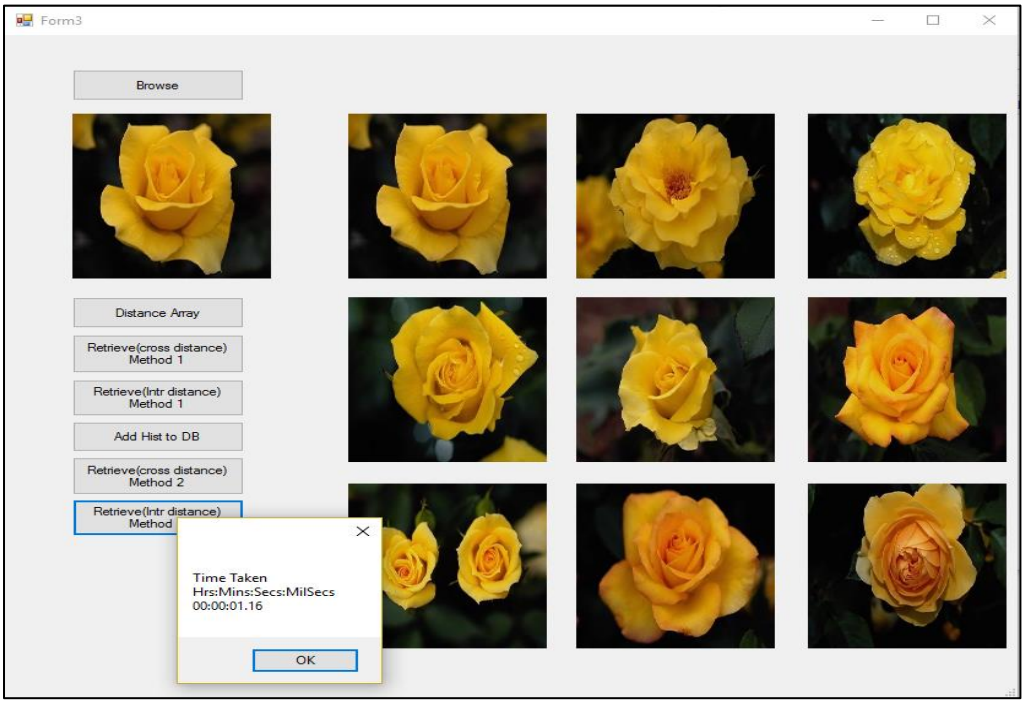

Figure 30. CBIR system–results set 6.

Two horses on a meadow.

Normalized cross correlation distance. Figure 31 shows the result set of the proposed content based image retrieval system, for a query image with two horses on a meadow. Normalized cross correlation distance used to retrieve following images.

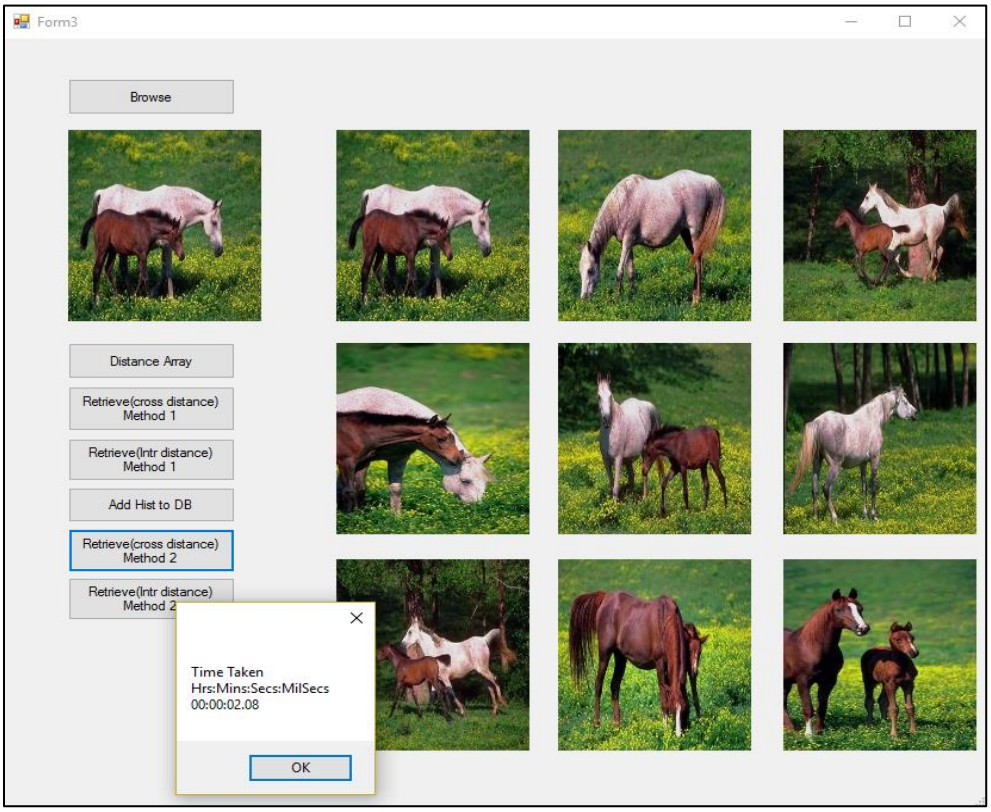

Figure 31. CBIR system–results set 7.

Histogram intersection distance. Figure 32 shows the result set of the proposed content based image retrieval system, for a query image with two horses on a meadow. Histogram intersection distance used to retrieve following images.

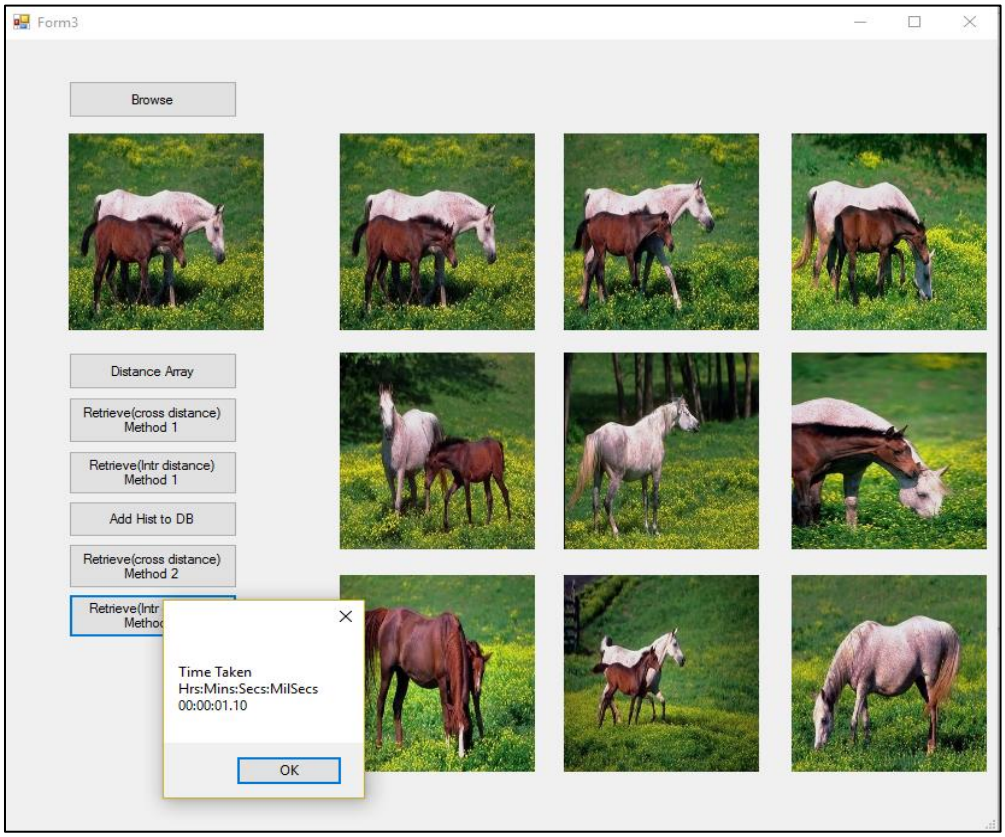

Figure 32. CBIR system–results set 8.

Chapter 5: CONCLUSION AND FUTURE WORK

All three distance measures are applicable. Similar images have a smaller value for the similarity measurement compared to dissimilar images. However, the distributions of those values are not similar for all methods. Both Euclidean distance and Histogram Intersection distance methods have a significant difference between similar images and dissimilar images compared to Normalized Cross Bin distance measure. By considering the gap between similar images and dissimilar images, it is concluded that Euclidean distance and Histogram Intersection distance methods are better. Between Euclidean distance and Histogram Intersection, Histogram Intersection distance measure has a big gap between similar images and dissimilar images. Therefore, Histogram Intersection Distance Methods is the most suitable distance measure for content based image retrieval System compared to other considered distance measures.

As discussed earlier, by comparing the accuracy of the taken distance measures, Histogram Intersection Distance turns out to be the best distance measure. Even if performance was taken into consideration, the Histogram Intersection Distance measure is faster than the Normalized Cross Correlation method. In addition, this research demonstrates that performance improvement can be conducted for the content based image retrieval system by storing intermediate results in the database instead of storing the full image.

This content based image retrieval system was implemented based on the color. The accuracy of the system can be enhanced by considering other visual features, textures, and shapes. In the future, other feature extraction methods can be fused with the color feature of this image retrieval system. In addition, this system might help content based image retrieval

systems build based on neural network. In the content based image retrieval system developed by Hanen, Mohamed, and Faiez, low-level feature extraction methods were utilized to generate feature vector and to train the neural network [13]. The data stored in this system's database will definitely help similar system.

To implement the content based image retrieval system 160 bins used. As mentioned in Chapter 3, bins count affect to both accuracy and performance of the system. An Image retrieval system with a large number of bins will give more accurate results, but it will slow down the process. On the other hand, a small number of bins will give images quickly, but it will not be accurate as the other system with a large number of bins. So optimal number of bins for color histogram distance need to be tested by changing the number of bins and evaluating the accuracy and the performance.

Since performance enhancement is main focus of this research security of the system did not take into consideration. But for any computer based system, security is a major concern. One of the vulnerable part of this system is the database. A pirate can access to the database system and can alter the database. The intruder can add some illegal materials to the database. Or that person can change the content or even can delete the data in the database. So it is clear that the security must be included to these systems. In the conference paper published by T.-T. Do, E. Kijak, T. Furon, and L. Amsaleg have explained how security can be breached in content based image retrieval systems. They performed three experiments to show how a pirate can reduce the recognition capabilities of the system [24].

Cloud computing can be used to enhance this content based image retrieval system. The data can be saved in a cloud. However with this enhancement the system can be more

vulnerable. To make these system more secure, researchers introduced several methods. For an example in the conference papers published by P. Saini, H. Singh, S. Lain, and S. Soni proposed a better cryptographic algorithm to for image retrieval systems [25].

References

[1] S. Deniziak and T. Michno, "Content based image retrieval using query by approximate shape," in *Proceedings of the Federated Conference on Computer Science and Information Systems*, 2016.

[2] T. John, "Techulator," 2012. [Online]. Available: http://www.techulator.com/resources/5933-What-Semantic-Search.aspx.

[3] H. Poulami and J. Mukherjee, "Content based image retrieval using histogram color and edge," *International Journal of Computer Applications* (0975-888), vol. 48, 11 June 2012.

[4] F. Malik and B. Baharudin, "Analysis of distance metric in content based image retrieval using statistical quantized histogram texture feature in the DCT domain," *Journal of King Saud University-Computer Sciences*, vol. 25, 02 July 2013.

[5] A. Anandh, K. Mala, and S. Suganya, "Content based image retrieval system based on semantic information using color, texture and shape feature," in *2016 International Conference on Computing Technologies and Intelligent Data Engineering*, 2016.

[6] J. Wu, "raywenderlich.com," July 2014. [Online]. Available: https://www.raywenderlich.com/69855/image-processing-in-ios-part-1-raw-bitmap-modification.

[7] S. Kaur and D. V. K. Banga, "Content based image retrieval survey and comparison between RGB and HSV model," *International Journal of Engineering Trends and Technology (IJETT)*, vol.4, no. 4, pp. 575-579, April 2013.

[8] N. Sharma, P. Rawat, and J. Singh, "Efficient CBIR using color histogram processing," *Signal & Image Processing: An International Journal (SIPIJ)*, vol. 2, no. 1, March 2011.

[9] J. M. Patel and N. C. Gamit, "A review on feature extraction techniques in content based image retrieval," in *IEEE WISPNET 2016 Conference*, 2016.

[10] Vadivel, Majumdar, and S. Sural, "Performance comparison of distance metrics in content-based image retrieval applications," in *International Conference on Information Technology (CIT), Bhubaneswar, India*, 2003.

[11] "http://docs.adaptive-vision.com/current/studio/machine_vision_guide/TemplateMatching.html," Adaptive Vision, 2016. [Online].

[12] M. D. Chaudhary and P. V. Pithadia, "Multi-feature histogram intersection for efficient content based image retrieval," in *International Conference on Circuit, Power and Computing Technologies,* 2014.

[13] H. Karamti, M. Tmar, and F. Gargouri, "Content-based image retrieval system using neural network," in *Computer Systems and Applications (AICCSA)*, Nov 2014.

[14] L. Pinjarkar, M. Sharma, and K. Mehta, "Comparison and analysis of content based image retrieval systems based on relevance feedback," *Journal of Emerging Trends in Computing and Information Sciences,* July 2012.

[15] H. Muller, W. Muller, D. M. Squire, S. Marchand-Maillet, and T. Pun, "Performance evaluation in content based image retrieval: overview and proposals," Dec 1999.

[16] M. J. Swain and D. H. Ballard, "Color indexing," *International Journal of Computer Vision,* 1991.

[17] F. Autrusseau, "http://www.irccyn.ec-nantes.fr," [Online].

[18] Pratuat, "SourceForge," [Online]. Available: https://sourceforge.net/projects/cbir-fyp/.

[19] G. K. Patro and K. K. Sahu, "Normalization: A preprocessing stage," *ResearchGate,* 2015.

[20] J. Li and J. Z. Wang, "James Z. Wang research group," [Online]. Available: http://wang.ist.psu.edu/docs/related/.

[21] J. Li and J. Z. Wang, "Automatic linguistic indexing of pictures by a statistical modeling approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 25, no 9, pp. 1075-1088, 2003.

[22] J. Z. Wang, J. Li, and G. Wiederhold, "SIMPLicity: Semantics-sensitive integrated matching for picture libraries," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* vol. 23, no. 9, pp. 947-963, 2001.

[23] Microsoft, "Microsoft developer network," Microsoft, 2016. [Online]. Available: https://msdn.microsoft.com/en-us/library/system.diagnostics.stopwatch(v=vs.110).aspx.

[24] T.-T. Do, E. Kijak, T. Furon and L. Amsaleg, "Challenginf the security of content_based image retrieval systems," in *Multimedia Signal Processing (MMSP),* October 2010.

[25] P. Saini, H. Singh, S. Lain, and S. Soni, "Image retrieval in cloud computing enviornment with the help of fuzzy semantic relevance matix," in *Computing for Sustainable Global Development(INDIACom)*, March 2016.

Appendix

Source Code–Content Based Image Retrieval System–Version 1

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using MySql.Data.MySqlClient;
using System.Diagnostics;

namespace CBIR
{
    public partial class Form3 : Form
    {
        Image file;
        Bitmap newBitmap, newBitmap2;
        public static int no_images = 999;
        double r, g, b;
        double h, s, v;
        double temp, min, sum;
        int hh, ss, vv;
        int[, ,] count = new int[10, 4, 4];
        double[] array1 = new double[10 * 4 * 4];
        double[] array2 = new double[10 * 4 * 4];
        string[] array3 = new string[10 * 4 * 4];
        double[] array4 = new double[10 * 4 * 4];
        double distance;
        double[,] dist = new double[2, no_images];
        double[] search = new double[no_images];
        double[] distSort = new double[no_images];
        int[] arrid = new int[9];
        String[,] arrdb = new String[2, no_images];
        String[] paths = new String[9];
        String[] histDb = new String[no_images];
        String string_hist;

        public Form3()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            DialogResult dr = openFileDialog1.ShowDialog();

            if (dr == DialogResult.OK)
            {
                file = Image.FromFile(openFileDialog1.FileName);
                newBitmap = new Bitmap(openFileDialog1.FileName);
                pictureBox1.Image = file;
```

```
        }
    }

    private int check_h(double h)
    {
        if (h >= 0 && h < 0.1)
            return 0;
        else if (h >= 0.1 && h < 0.2)
            return 1;
        else if (h >= 0.2 && h < 0.3)
            return 2;
        else if (h >= 0.3 && h < 0.4)
            return 3;
        else if (h >= 0.4 && h < 0.5)
            return 4;
        else if (h >= 0.5 && h < 0.6)
            return 5;
        else if (h >= 0.6 && h < 0.7)
            return 6;
        else if (h >= 0.7 && h < 0.8)
            return 7;
        else if (h >= 0.8 && h < 0.9)
            return 8;
        else
            return 9;
    }


    private int check_s(double s)
    {
        if (s >= 0 && s < 0.25)
            return 0;
        else if (s >= 0.25 && s < 0.5)
            return 1;
        else if (s >= 0.5 && s < 0.75)
            return 2;
        else
            return 3;
    }


    private int check_v(double v)
    {
        if (v >= 0 && v < 0.25)
            return 0;
        else if (v >= 0.25 && v < 0.5)
            return 1;
        else if (v >= 0.5 && v < 0.75)
            return 2;
        else
            return 3;

    }
```

```
        public double[] hist(int[, ,] count)
        {

            double[] array = new double[10 * 4 * 4];
            int bincount = 0;

            for (int i = 0; i < 10; i++)
                for (int j = 0; j < 4; j++)
                    for (int k = 0; k < 4; k++)
                    {

                        array[bincount] = count[i, j, k];
                        bincount++;

                    }

            return array;

        }

        public int[, ,] pro(Bitmap newBitmap)
        {
            for (int x = 0; x < newBitmap.Width; x++)
            {
                for (int y = 0; y < newBitmap.Height; y++)
                {
                    Color pixel = newBitmap.GetPixel(x, y);

                    r = pixel.R;
                    g = pixel.G;
                    b = pixel.B;

                    if (r == g && g == b)
                    {
                        h = 0;
                        s = 0;
                        v = r;
                    }

                    else
                    {
                        temp = r * r + g * g + b * b - (r * g + g * b + r * b);

                        temp = (2 * r - g - b) / (2 * Math.Sqrt((double)temp));

                        h = Math.Acos((double)temp) / Math.PI;

                        min = r;
                        if (min > g)
                            min = g;
                        if (min > b)
                            min = b;
                        sum = r + g + b;
                        s = 1 - (3 * min) / sum;
                        v = sum / 3;
```

```
                    // H, S & V between 0.0-1.0
                }

                // HSV -> 10*4*4= 160 quantization bins

                hh = check_h(h);
                ss = check_s(s);
                vv = check_v(v);
                count[hh, ss, vv]++;
            }
        }

        sum = 0;
        for (int i = 0; i < 10; i++)
            for (int j = 0; j < 4; j++)
                for (int k = 0; k < 4; k++)
                {
                    sum = sum + count[i, j, k];
                }

        return count;
    }



    private void button4_Click(object sender, EventArgs e)
    {
        //Implementing a Stop Watch to Track Time
        Stopwatch stopWatch = new Stopwatch();
        stopWatch.Start();

        String connstring = "server=localhost;database=dbcbir4;uid=root";
        MySqlConnection conn = new MySqlConnection(connstring);
        MySqlCommand command = conn.CreateCommand();

        command.CommandText = "select * from images;";

        try
        {
            conn.Open();
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
        }

        MySqlDataReader reader = command.ExecuteReader();


        int cnt = 0;

        Form3 c = new Form3();
        count = c.pro(newBitmap);
        array1 = c.hist(count);
        for (int i = 0; i < 10; i++)
```

```
                for (int j = 0; j < 4; j++)
                    for (int k = 0; k < 4; k++)
                    {
                        count[i, j, k] = 0;
                    }

            int num = 0;
            while (reader.Read())
            {
                arrdb[0, cnt] = reader["imgid"].ToString();
                arrdb[1, cnt] = reader["imgpath"].ToString();
                if (cnt == num)
                {

                    file = Image.FromFile(reader["imgpath"].ToString());
                    newBitmap2 = new Bitmap(reader["imgpath"].ToString());

                    count = c.pro(newBitmap2);
                    array2 = c.hist(count);
                    for (int i = 0; i < 10; i++)
                        for (int j = 0; j < 4; j++)
                            for (int k = 0; k < 4; k++)
                            {
                                count[i, j, k] = 0;
                            }

                    Intr id = new Intr();
                    double distance = id.intrdist(array1, array2);
                    dist[0, cnt] = cnt + 1;
                    dist[1, cnt] = Math.Round(distance, 6);
                    dist[1, cnt] = Math.Abs(dist[1, cnt]);
                    distSort[cnt] = Math.Round(distance, 6);
                    distSort[cnt] = Math.Abs(distSort[cnt]);
                    search[cnt] = Math.Round(distance, 6);
                    search[cnt] = Math.Abs(search[cnt]);

                }
                cnt++;
                num++;
            }

            Array.Sort(distSort);

            int index = distSort.Length;

            for (int k = 0; k < 9; k++)
            {
                for (int i = 0; i < index; i++)
                {
                    if (dist[1, i] == distSort[k])
                    {
                        arrid[k] = (int)dist[0, i];
                        break;
                    }
                }
```

```
            }

            conn.Close();

            for (int k = 0; k < 9; k++)
            {
                for (int i = 0; i < no_images; i++)
                {
                    if (arrdb[0, i] == arrid[k].ToString())
                    {
                        paths[k] = arrdb[1, i];
                        break;
                    }
                }
            }

            pictureBox2.Image = Image.FromFile(paths[0]);
            pictureBox3.Image = Image.FromFile(paths[1]);
            pictureBox4.Image = Image.FromFile(paths[2]);
            pictureBox5.Image = Image.FromFile(paths[3]);
            pictureBox6.Image = Image.FromFile(paths[4]);
            pictureBox7.Image = Image.FromFile(paths[5]);
            pictureBox8.Image = Image.FromFile(paths[6]);
            pictureBox9.Image = Image.FromFile(paths[7]);
            pictureBox10.Image = Image.FromFile(paths[8]);

            stopWatch.Stop();

            //Get the elapsed time as a TimeSpan value
            TimeSpan ts = stopWatch.Elapsed;

            //Format and display the TimeSpan value.
            String elapsedTime = String.Format("{0:00}:{1:00}:{2:00}.{3:00}",
            ts.Hours, ts.Minutes, ts.Seconds,
            ts.Milliseconds / 10);
            MessageBox.Show("Time Taken\nHrs:Mins:Secs:MilSecs\n" + elapsedTime);
        }

    }

}
```

Intr.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace CBIR
{
    class Intr
```

```
    {
        public double intrdist(double[] array1,double[] array2)
        {

            if (array2.Length == array1.Length)
            {
                double sum1, sum2, sum3;
                sum1 = 0.0;
                sum2 = 0.0;
                sum3 = 0.0;

                for (int i = 0; i < array2.Length; i++)
                {
                         if (array1[i] <= array2[i])
                             sum1 = sum1 + array1[i];
                         else
                             sum1 = sum1 + array2[i];

                        sum2+=array1[i];
                        sum3+=array2[i];
                }

                if (sum2 <= sum3)
                    return (1-sum1 / sum2);
                else
                    return (1-sum1 / sum3);
            }
            else
            {
                return -1;
            }
        }
    }
}
```

Source code – New method with changes to database

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using MySql.Data.MySqlClient;
using System.Diagnostics;

namespace CBIR
{
    public partial class Form3 : Form
```

```
    {
        Image file;
        Bitmap newBitmap, newBitmap2;
        public static int no_images = 999;
        double r, g, b;
        double h, s, v;
        double temp, min, sum;
        int hh, ss, vv;
        int[, ,] count = new int[10, 4, 4];
        double[] array1 = new double[10 * 4 * 4];
        double[] array2 = new double[10 * 4 * 4];
        string[] array3 = new string[10 * 4 * 4];
        double[] array4 = new double[10 * 4 * 4];
        double distance;
        double[,] dist = new double[2, no_images];
        double[] search = new double[no_images];
        double[] distSort = new double[no_images];
        int[] arrid = new int[9];
        String[,] arrdb = new String[2, no_images];
        String[] paths = new String[9];
        String[] histDb = new String[no_images];
        String string_hist;

        public Form3()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            DialogResult dr = openFileDialog1.ShowDialog();

            if (dr == DialogResult.OK)
            {
                file = Image.FromFile(openFileDialog1.FileName);
                newBitmap = new Bitmap(openFileDialog1.FileName);
                pictureBox1.Image = file;

            }
        }

        private int check_h(double h)
        {
            if (h >= 0 && h < 0.1)
                return 0;
            else if (h >= 0.1 && h < 0.2)
                return 1;
            else if (h >= 0.2 && h < 0.3)
                return 2;
            else if (h >= 0.3 && h < 0.4)
                return 3;
            else if (h >= 0.4 && h < 0.5)
                return 4;
            else if (h >= 0.5 && h < 0.6)
                return 5;
```

```
            else if (h >= 0.6 && h < 0.7)
                return 6;
            else if (h >= 0.7 && h < 0.8)
                return 7;
            else if (h >= 0.8 && h < 0.9)
                return 8;
            else
                return 9;
        }


        private int check_s(double s)
        {
            if (s >= 0 && s < 0.25)
                return 0;
            else if (s >= 0.25 && s < 0.5)
                return 1;
            else if (s >= 0.5 && s < 0.75)
                return 2;
            else
                return 3;
        }


        private int check_v(double v)
        {
            if (v >= 0 && v < 0.25)
                return 0;
            else if (v >= 0.25 && v < 0.5)
                return 1;
            else if (v >= 0.5 && v < 0.75)
                return 2;
            else
                return 3;

        }

        public double[] hist(int[, ,] count)
        {

            double[] array = new double[10 * 4 * 4];
            int bincount = 0;

            for (int i = 0; i < 10; i++)
                for (int j = 0; j < 4; j++)
                    for (int k = 0; k < 4; k++)
                    {

                        array[bincount] = count[i, j, k];
                        bincount++;

                    }

            return array;
```

```
        }

        public int[, ,] pro(Bitmap newBitmap)
        {
            for (int x = 0; x < newBitmap.Width; x++)
            {
                for (int y = 0; y < newBitmap.Height; y++)
                {
                    Color pixel = newBitmap.GetPixel(x, y);

                    r = pixel.R;
                    g = pixel.G;
                    b = pixel.B;

                    if (r == g && g == b)
                    {
                        h = 0;
                        s = 0;
                        v = r;
                    }

                    else
                    {
                        temp = r * r + g * g + b * b - (r * g + g * b + r * b);

                        temp = (2 * r - g - b) / (2 * Math.Sqrt((double)temp));

                        h = Math.Acos((double)temp) / Math.PI;

                        min = r;
                        if (min > g)
                            min = g;
                        if (min > b)
                            min = b;
                        sum = r + g + b;
                        s = 1 - (3 * min) / sum;
                        v = sum / 3;
                        // H, S & V between 0.0-1.0
                    }

                    // HSV -> 10*4*4= 160 quantization bins

                    hh = check_h(h);
                    ss = check_s(s);
                    vv = check_v(v);
                    count[hh, ss, vv]++;
                }
            }

            sum = 0;
            for (int i = 0; i < 10; i++)
                for (int j = 0; j < 4; j++)
                    for (int k = 0; k < 4; k++)
                    {
                        sum = sum + count[i, j, k];
```

```
                }

            return count;
        }

        private void button5_Click(object sender, EventArgs e)
        {
            String connstring = "server=localhost;database=dbcbir4;uid=root;";
            MySqlConnection conn = new MySqlConnection(connstring);
            MySqlCommand command = conn.CreateCommand();

            command.CommandText = "select * from images;";

            try
            {
                conn.Open();
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex.Message);
            }

            MySqlDataReader reader = command.ExecuteReader();


            int cnt = 0;

            Form3 c = new Form3();
            count = c.pro(newBitmap);
            array1 = c.hist(count);
            for (int i = 0; i < 10; i++)
                for (int j = 0; j < 4; j++)
                    for (int k = 0; k < 4; k++)
                    {
                        count[i, j, k] = 0;
                    }

            int num = 0;
            int m = 0;


            while (reader.Read())
            {
                arrdb[0, cnt] = reader["imgid"].ToString();
                arrdb[1, cnt] = reader["imgpath"].ToString();
                if (cnt == num)
                {

                    file = Image.FromFile(reader["imgpath"].ToString());
                    newBitmap2 = new Bitmap(reader["imgpath"].ToString());

                    count = c.pro(newBitmap2);
                    array2 = c.hist(count);

                    int l = array1.Length;
```

```
                    String z = l.ToString();

                    string output = " ";


                    for (int k = 0; k < 160; k++)
                    {
                        output += array2[k].ToString() + " ";
                    }

                     histDb[m] = output;

                    m++;

                    for (int i = 0; i < 10; i++)
                        for (int j = 0; j < 4; j++)
                            for (int k = 0; k < 4; k++)
                            {
                                count[i, j, k] = 0;
                            }

                }
                cnt++;
                num++;

            }


            conn.Close();

            try
            {
                conn.Open();
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex.Message);
            }

            for (int q = 500; q < no_images; q++)
            {
                command.CommandText = "update images set hist='" + histDb[q] + "'
where imgid=" + (q + 1) + ";";
                command.ExecuteNonQuery();
            }

            /*for(int q = 501; q < 1001; q++)
            {
                command.CommandText = "update images set
imgpath='d:\\\\imageDB2\\\\" + q + ".jpg' where imgid=" + q + ";";
                command.ExecuteNonQuery();
            }*/
            conn.Close();
```

```
        }


        private void button7_Click(object sender, EventArgs e)
        {

            //Implementing a Stop Watch to Track Time
            Stopwatch stopWatch = new Stopwatch();
            stopWatch.Start();

            String connstring = "server=localhost;database=dbcbir4;uid=root";
            MySqlConnection conn = new MySqlConnection(connstring);
            MySqlCommand command = conn.CreateCommand();

            command.CommandText = "select * from images;";

            try
            {
                conn.Open();
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex.Message);
            }

            MySqlDataReader reader = command.ExecuteReader();


            int cnt = 0;

            Form3 c = new Form3();
            count = c.pro(newBitmap);
            array1 = c.hist(count);
            for (int i = 0; i < 10; i++)
                for (int j = 0; j < 4; j++)
                    for (int k = 0; k < 4; k++)
                    {
                        count[i, j, k] = 0;
                    }

            int num = 0;
            while (reader.Read())
            {
                arrdb[0, cnt] = reader["imgid"].ToString();
                arrdb[1, cnt] = reader["imgpath"].ToString();
                if (cnt == num)
                {

                    file = Image.FromFile(reader["imgpath"].ToString());

                    string_hist = reader["hist"].ToString();

                    //Split numbers
```

```
                    array3 = string_hist.Split(new string[] { " " },
StringSplitOptions.None);

                    for (int k = 0; k < 160; k++)
                    {
                        //Convert string to int
                        array4[k] = Double.Parse(array3[k + 1]);
                    }

                    for (int i = 0; i < 10; i++)
                        for (int j = 0; j < 4; j++)
                            for (int k = 0; k < 4; k++)
                            {
                                count[i, j, k] = 0;
                            }

                    Intr id = new Intr();

                    double distance=id.intrdist(array1,array4);
                    dist[0, cnt] = cnt + 1;
                    dist[1, cnt] = Math.Round(distance, 6);
                    dist[1, cnt] = Math.Abs(dist[1, cnt]);
                    distSort[cnt] = Math.Round(distance, 6);
                    distSort[cnt] = Math.Abs(distSort[cnt]);
                    search[cnt] = Math.Round(distance, 6);
                    search[cnt] = Math.Abs(search[cnt]);

                }
                cnt++;
                num++;
            }

            Array.Sort(distSort);

            int index = distSort.Length;


            for (int k = 0; k < 9; k++)
            {
                for (int i = 0; i < index; i++)
                {
                    if (dist[1, i] == distSort[k])
                    {
                        arrid[k] = (int)dist[0, i];
                        break;
                    }
                }
            }

            conn.Close();

            for (int k = 0; k < 9; k++)
            {
                for (int i = 0; i < no_images; i++)
                {
```

```
                    if (arrdb[0, i] == arrid[k].ToString())
                    {
                        paths[k] = arrdb[1, i];
                        break;
                    }
                }
            }

            pictureBox2.Image = Image.FromFile(paths[0]);
            pictureBox3.Image = Image.FromFile(paths[1]);
            pictureBox4.Image = Image.FromFile(paths[2]);
            pictureBox5.Image = Image.FromFile(paths[3]);
            pictureBox6.Image = Image.FromFile(paths[4]);
            pictureBox7.Image = Image.FromFile(paths[5]);
            pictureBox8.Image = Image.FromFile(paths[6]);
            pictureBox9.Image = Image.FromFile(paths[7]);
            pictureBox10.Image = Image.FromFile(paths[8]);


            stopWatch.Stop();

            //Get the elapsed time as a TimeSpan value
            TimeSpan ts = stopWatch.Elapsed;

            //Format and display the TimeSpan value.
            String elapsedTime = String.Format("{0:00}:{1:00}:{2:00}.{3:00}",
            ts.Hours, ts.Minutes, ts.Seconds,
            ts.Milliseconds / 10);
            MessageBox.Show("Time Taken\nHrs:Mins:Secs:MilSecs\n" + elapsedTime);
        }

    }

}
```

Intr.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace CBIR
{
    class Intr
    {
        public double intrdist(double[] array1,double[] array2)
        {

            if (array2.Length == array1.Length)
            {
                double sum1, sum2, sum3;
```

```
                sum1 = 0.0;
                sum2 = 0.0;
                sum3 = 0.0;

                for (int i = 0; i < array2.Length; i++)
                {
                         if (array1[i] <= array2[i])
                             sum1 = sum1 + array1[i];
                         else
                             sum1 = sum1 + array2[i];

                        sum2+=array1[i];
                        sum3+=array2[i];
                }

                if (sum2 <= sum3)
                    return (1-sum1 / sum2);
                else
                    return (1-sum1 / sum3);
            }
            else
            {
                return -1;
            }
        }
    }
}
```