ELE725

Content-Based Image Retrieval

Steve Singh *(Lead Author)*
Ryerson University
Department of Electrical and Computer Engineering
Toronto, Canada
steve.singh@ryerson.ca

***Abstract*— Content based Image retrieval was performed by comparing three channel histograms of images to the histogram of a sample query image. The histograms were created using RGB and HSV colour spaces. The similarity metrics calculated in HSV gave more perceptually accurate results, as it allows for much more better colour distinction when compared to RGB. When the difference metrics were compared, the Euclidean distance in the HSV colour space performed the best for query images inside the database. However, when querying the database with the chosen external image, using the Manhattan distance inside the RGB colour space provided the most accurate similarity ordering.**

Keywords-CBIR; RGB; HSV; histogram; similarity; Manhattan; Euclidean; Histogram Intersection;

I. INTRODUCTION

The purpose of the lab was to explore some concepts related to content-based image retrieval (CBIR). Namely, using histograms as content descriptors and various similarity metric. During the lab, a set of images were compared to a single query image using CBIR. The technique involved the acquisition of histograms in two colour spaces: RGB and HSV. This was done for of all of the images. The histogram of each of the sample image was then compared and sorted in terms of similarity with the query. Finally, the results were used to make comparisons as to which colour space and similarity metric combinations were best.

II. THEORY

Basic template matching can be achieved via template matching with colour histograms as the content descriptors [1]. Analysis can be performed by utilizing both RGB and HSV histograms. The color of each pixel in an RGB image is broken down into three color channels: Red, Green, and Blue. In the HSV color space, the color is described by its hue, saturation, and value (similar to luminance). The histograms of these color channels can be concatenated to create a one-dimensional histogram, making it easier to perform similarity checks using various difference measures. Among the difference measures are Manhattan distance, Euclidean distance, and histogram intersection (see equations 1-3).

Manhattan distance

$$d_{L1}(h,g) = \sum_i |h(i) - g(i)| \quad (1)$$

Euclidean distance

$$d_{L2}(h,g) = \sum_i (h(i) - g(i))^2 \quad (2)$$

Histogram intersection

$$d_{int}(h,g) = \frac{\sum_i \min(h(i), g(i))}{\min(|h|, |g|)} \quad (3)$$

Both Manhattan and Euclidean difference calculations, described by equations (1) and (2), rely on determining the difference between the histograms. The histogram intersection calculated by equation (3), on the other hand, only takes into the account the parts of the histogram which are shared.

III. METHODOLOGY

Content-Based Image Retrieval (CBIR)

The first step in creating the CBIR system was to gather 20 images, all named '1…20.jpg' for convenience. After the query image is selected, its histograms in the RGB and HSV colour spaces were concatenated for each of the three channels. This was done by writing a function in MATLAB called HistConcat(), which accepted the image signal, colour space (passing 1 = RGB, 2 = HSV), and number of bins:

```
function [ hist ] = HistConcat(img,
colour_space, bins)
  % Convert image to HSV
  % if specified
  if colour_space == 2
    img = rgb2hsv(img);
  end
```

If the color space is specified to be HSV (a value of 2), then the image is converted to HSV using the built-in rgb2hsv() MATLAB function, otherwise the RGB colour space is used by default. The next step was to obtain the histograms for each of the three channels (whether it was R, G, and B or H, S, and V) with usage of the imhist() function, which returns a column vector histogram. Finally, the joined histogram containing each channel is obtained by transposing the three column vectors and concatenating them as seen below:

```
    % Create 3 histograms by
    % using each channel
    A = imhist(img(:,:,1),bins);
    B = imhist(img(:,:,2),bins);
    C = imhist(img(:,:,3),bins);

    hist = [A' B' C'];
  end
```

Next, an index of RGB and HSV histograms for each image in the collection was created by writing a function named IndexImages(), which required passing the image directory and the query image number. The images were first placed into an array as follows:

```
function [ RGB_h, HSV_h, images ] =
IndexImages(directory, query)
  % Create array to store RGB histograms
  RGB_h = [];
  % Create array to store HSV histograms
  HSV_h = [];
  % Collect all images in image folder
  % and store inside an array
  images = {};

  % Load image directory
  d = dir(directory);



  % Delete directory entry containing
  % the query image to avoid compari-
  % sons with itself
  for i=1:length(d)
    if strcmp(d(i).name,query) == 1
      d(i) = []; break;
    end
  end

  for i = 1:length(d)
    % Ignore query image
    images{i} =
imread(strcat('images/',d(i).name));
  end
```

After storing all images (except for the query image) inside of the images array, the histograms are created for each of them using the RGB and HSV colour spaces:

```
    for i = 1:length(images)
      % Combine into RGB histogram
      RGB_h(i,:) =
    HistConcat(images{i},1,150);
      % Combine into HSV histogram
      HSV_h(i,:) =
    HistConcat(images{i},2,150);
    end
  end
```

Once the histograms were created and stored, the difference measures (using eq. 1-3) were performed in a function named Differences() which took two histograms and returned the manhattan distance between them, the Euclidean distance, as well as the histogram intersection value (see table 1). Before the differences could be calculated, MATLAB's repmat() function was used to repeat the histogram of the query image and create a matrix equal to the size of the RGB or HSV histogram matrices created with the above IndexImages() function:

```
function [MD,ED,HI] =
Differences(hist,q_hist)
  MD = []; ED = []; HI = [];
  [m,n] = size(hist);
  Q = repmat(q_hist,m,1);
  % Load matrix of histograms
  for i = 1:m
    X(i,:) = hist(i,:);
  end
  % Calculate Manhattan distance
    MD = sum(abs(Q - X),2);
  % Calculate Euclidean distance
    ED = sqrt(sum((Q - X).^2,2));
  % Calculate Hist. intersection
    HI =
    sum(min(Q,X)/min(abs(Q),abs(X)),2);
end
```

In the main script, the aforementioned functions are used as follows:

```
% Set image directory
directory = 'images/*.jpg';
% Set query image
query = '1.jpg';
q_file = strcat('images/',query);
q = imread(q_file);
q_RGB = HistConcat(q,1,150);
q_HSV = HistConcat(q,2,150);
% Retrieve index of RGB and
% HSV histograms
[RGB_h, HSV_h, images] =
IndexImages(directory, query);

% Perform difference measures
[MD1, ED1, HI1] = Differences(RGB_h,
q_RGB);
% Repeat for HSV histograms
```

```
[MD2, ED2, HI2] = Differences(HSV_h,
q_HSV);
```

After obtaining the Manhattan distances, Euclidean distances, and histogram intersection values, they needed to be sorted. However, a challenge did arise: since the values were returned in the form of column vectors, there was no direct way of associating each value with an image after sorting. To remedy this, the indices were added as another column in each column vector, allowing the images to be referenced after sorting:

```
[m,n] = size(MD1);
for i = 1:m
  % Add indices for tracking
  % after sorting
  MD1(i,2) = i;
  ED1(i,2) = i;
  HI1(i,2) = i;
  MD2(i,2) = i;
  ED2(i,2) = i;
  HI2(i,2) = i;
end

% Sort difference measures
MD1 = sortrows(MD1,1);
ED1 = sortrows(ED1,1);
HI1 = sortrows(HI1,1);
MD2 = sortrows(MD2,1);
ED2 = sortrows(ED2,1);
HI2 = sortrows(HI2,1);
```

After sorting based on the first column, the images were displayed according to the second column of each difference vector (note that the code for displaying the similarity in the RGB colour space with manhattan distance is shown below; other colour space and difference measure combinations were displayed in a similar manner) :

```
% Display images by similarity
% based on above measures
subplot(5,4,1);
imshow(q); title('Query Image');
for i = 1:m
  subplot(5,4,i+1);
  imshow(images{MD1(i,2)});
end
figure;
```

IV. Results

A. *Image Similarity in RGB and HSV colour spaces*

The query image can be seen in figure 1 below.

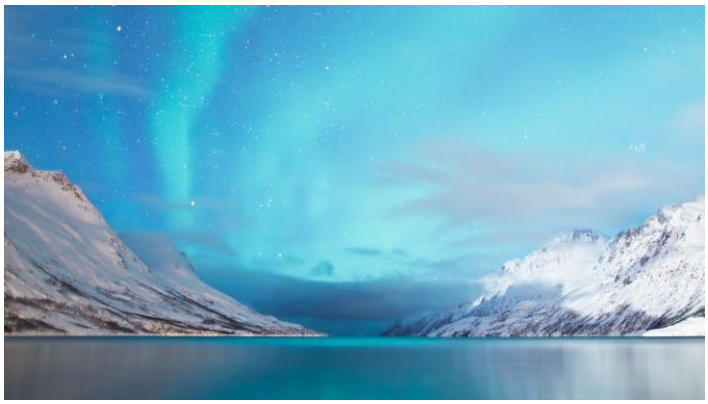

Figure 1: Image used to query the collection

Figure 2 shows the similarity ranking (left to right from high to low) of the query image in the RGB colour space using the Manhattan distance measure. The result can be seen in more detail with the included figures. Note that the number of bins used for each channel was 150.

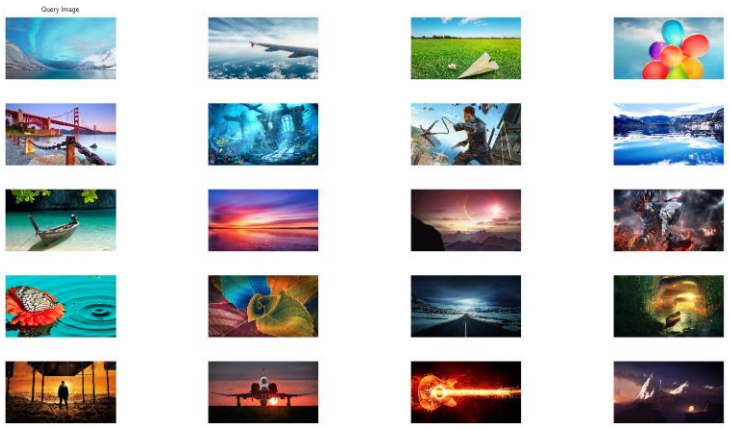

Figure 2: Image similarity ranking using Manhattan distance in the RGB colour space

Similarly, figure 3 shows the image similarity in the RGB colour space using Euclidean distance as the difference measure. Figure 4 shows the rankings in the RGB colour space after performing histogram intersection.

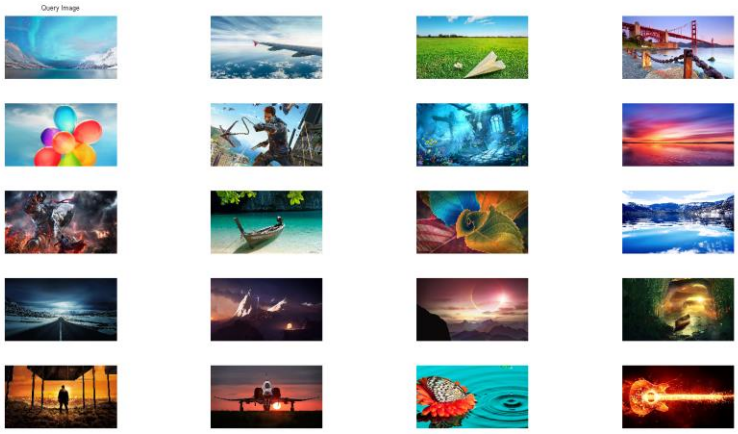

Figure 3: Image similarity ranking using Euclidean distance in the RGB colour space

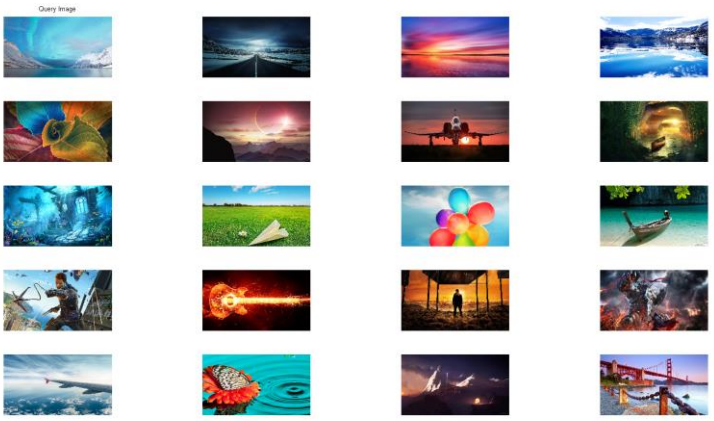

Figure 4: Image similarity ranking using histogram intersection in the RGB colour space

In the HSV colour space, figures 5, 6, and 7 show the similarity ranking using the Manhattan distance, Euclidean distance, and histogram intersection (respectively).

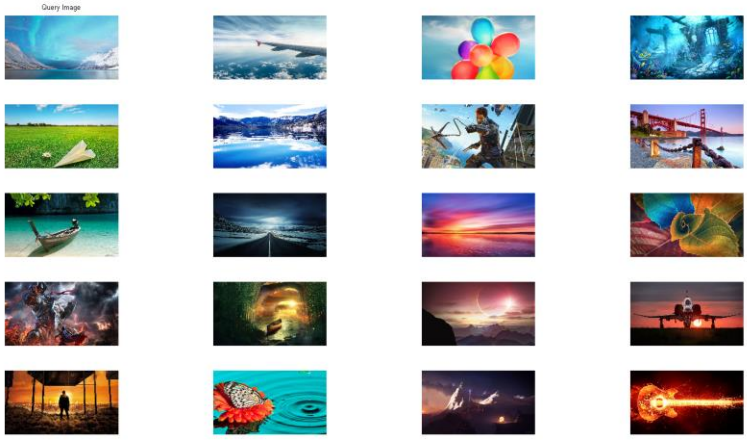

Figure 5: Image similarity ranking using Manhattan distance in the HSV colour space

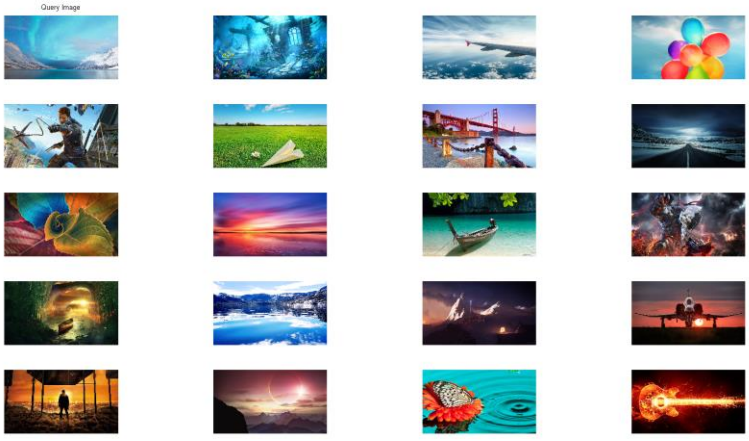

Figure 6: Image similarity ranking using Euclidean distance in the HSV colour space

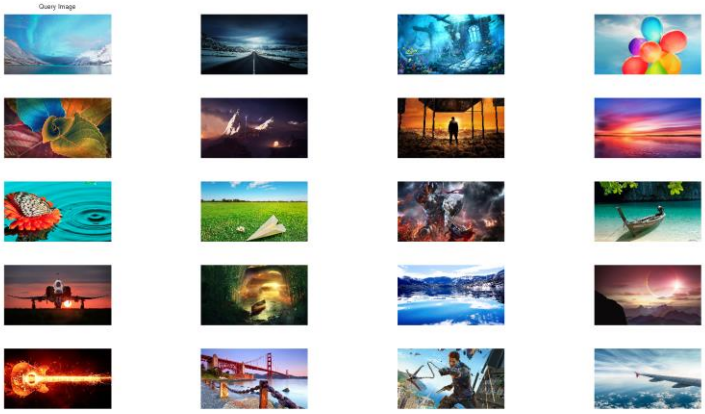

Figure 7: Image similarity ranking using histogram intersection in the HSV colour space

The difference measures for each colour space and difference measure combination is shown in tables 1-6 below:

Image	Value	Image	Value
2.jpg	2583802	14.jpg	2217602
3.jpg	2359796	15.jpg	1998454
4.jpg	4069544	16.jpg	2772658
5.jpg	3209784	17.jpg	4042046
6.jpg	2803936	18.jpg	3891556
7.jpg	3912342	19.jpg	4178964
8.jpg	3169162	20.jpg	3805426
9.jpg	3140692		
10.jpg	2599996		
11.jpg	3088416		
12.jpg	1930856		
13.jpg	1527014		

Table 1: Manhattan distance measures in RGB colour space

Image	Value	Image	Value
2.jpg	1.9439×10^5	12.jpg	1.4840×10^5
3.jpg	1.9808×10^5	13.jpg	1.2317×10^5
4.jpg	6.3949×10^5	14.jpg	1.5690×10^5
5.jpg	2.4534×10^5	15.jpg	1.7780×10^5
6.jpg	2.1439×10^5	16.jpg	2.3532×10^5
7.jpg	2.9249×10^5	17.jpg	3.2065×10^5
8.jpg	4.7760×10^5	18.jpg	2.9072×10^5
9.jpg	2.2101×10^5	19.jpg	2.8697×10^5
10.jpg	2.6357×10^5	20.jpg	2.8400×10^5
11.jpg	2.8700×10^5		

Table 2: Euclidean distance measures in RGB colour space

Image	Value	Image	Value
2.jpg	2632900	14.jpg	2779478
3.jpg	2259268	15.jpg	1980282
4.jpg	4590198	16.jpg	2956658
5.jpg	3266848	17.jpg	3644926
6.jpg	3147404	18.jpg	3580676
7.jpg	3798894	19.jpg	4143030
8.jpg	3962426	20.jpg	2976302
9.jpg	3282352		
10.jpg	2600444		
11.jpg	3638524		
12.jpg	2426364		
13.jpg	1772066		

Table 3: Manhattan difference results in HSV colour space

Image	Value	Image	Value
2.jpg	2.2341×10^5	14.jpg	2.8352×10^5
3.jpg	2.1135×10^5	15.jpg	2.2115×10^5
4.jpg	7.0094×10^5	16.jpg	3.1335×10^5
5.jpg	2.9483×10^5	17.jpg	3.6964×10^5
6.jpg	3.1155×10^5	18.jpg	3.2308×10^5
7.jpg	3.7742×10^5	19.jpg	3.5274×10^5
8.jpg	6.8515×10^5	20.jpg	2.9374×10^5
9.jpg	3.1390×10^5		
10.jpg	3.4437×10^5		
11.jpg	3.9268×10^5		
12.jpg	2.6099×10^5		
13.jpg	2.1755×10^5		

Table 4: Euclidean distance measure in HSV colour space

Please note that for the histogram intersection values, MATLAB displayed 1.0000 for all, but the values were not all identical and were of higher precision than what was displayed. Their values were sorted accordingly using the sortrows() function. The tables for histogram intersection were not included for this reason.

V. Discussion

The results of the content-based image retrieval shows that when using 150 bins for each channel, the similarity rankings are most perceptually accurate when using the Euclidean difference measure in the HSV colour space. It was expected that one of the difference measures in conjunction with the HSV colour space would result in more accurate image comparisons since using HSV is better for distinguishing based on colour [2].

Calculating the similarity metrics in the RGB colour space does not always provide perceptually accurate results, since it makes comparisons purely based on the values of each chroma channel. For example, if a comparison were made between a white and green region of an image, there would be some similarity detected since white requires a value of 255 from the green channel. Perceptually speaking, however, green and white are very different in appearance.

Increasing the number of bins for the H channel provided more accurate results in the similarity rankings for each image in the collection. This is likely due to the fact that the H channel (hue) in the HSV colour space contains most of the colour information [2]. Interestingly, when querying with an image from outside of the 20-image database (’99.jpg’), the best results are seen using the Manhattan distance similarity metric with the RGB colour space.

VI. Conclusion

Content-based image retrieval was performed during the lab. An image was used as a query to an image database with the aim of determining the order in which they are similar. The combined three-channel histogram of the query image was compared with that of each image in the database using three similarity metrics: Manhattan distance, Euclidean distance, and histogram intersection. Furthermore, two colour spaces were used: RGB and HSV. It was found that the Euclidean distance in conjunction with the HSV colour space provided the best perceptual accuracy in terms of ranking by similarity. However, when an image from outside of the database was used, it was found that using the Manhattan distance measure inside the RGB colour space was most accurate. Overall, the general observation was that using the HSV colour space tends to provide more perceptual accuracy because it allows for better colour distinction than RGB.

REFERENCES

[1] Department of Engineering and Architecture. (2014). *Motion Compensation/CBIR* [Lab Manual]. Toronto, ON: Ryerson University.

Kyan M. (2013). *Content-based Analysis[1]* [PDF document]. Retrieved from https://courses.ryerson.ca/bbcswebdav/pid-2677267-dt-content-rid-2670666_2/courses/ele725_f12_01/ELE725_F2012__L10_ContentBasedAnalysis.pdf