SCC0251 - Image Processing
Prof. Moacir Ponti

- Final Report -

Search for similar images from a database
Group 19

Frederico de Oliveira Sampaio - 8922100
Julia Minetto Macias - 8937329
Instituto de Ciências Matemáticas e de Computação
USP São Carlos

Summary

1. Preface

The project developed refers to the Image Processing project and aims to develop a content image comparison system the should return the *n* most similar images related to the input image. All the processes consist in implementing image descriptors to extract image characteristics according to their color, shapes, etc. We are using a image database with a thousand images contemplating many different categories. The comparison is made using the concept of DFT (Discrete Fourier Transform), GCH (Global Color Histogram) and LCH (Local Color Histogram).

2. Input images

The input image it's up to the user. We used, as a test for our project, 1000 random images that were found on a website, specified at the end of this document.

We expect finding photos with the same "theme" or characteristics.

3. Process Details

During the project development it was used several concepts and techniques about image processing learned in class and some of them searched externally. To extract the characteristics vector of the images it was implemented the Fourier descriptor and the global and local histogram color descriptor. Worth to emphasizing that when using Fourier, the images are used in grayscale, and when using color descriptors, the images are converted to HSV (Hue/Saturation/Value) model.

The index to all methods is generated by extracting the characteristic vector of the images database and stored in *.csv* file which each line represents a characteristics vector of some image. The first column is the image id and the other many columns is a characteristic number.

The images search in the image database is done through the extraction of characteristics of the query image and the respective comparison of its characteristic vector with the characteristic vectors of the index. This comparison is made by a distance metric that can be Chi-Square, Euclidean or Manhattan.Such metric compares the similarity of two characteristic vectors so that the smaller distance between them, the more similar the images are.

4. Development Detail

4.1. Implemented Descriptors

In this application it was used three image descriptors, being two color ones and the fourier thee other one.

4.1.1. Fourier Descriptor

The Discrete Fourier Transform extracts characteristics related on the images shapes. It represents the spatial domain in the frequency domain. Knowing the high computing cost this method has, the transform is using masks, so we can find very similar results with a lower computing cost.

4.1.2 Color Descriptor - Global Color Histogram

The global color histogram descriptor extracts the full image histogram for each channel in HSV model. The HSV channels was divided in appropriate intervals detailed:

-The *Hue* channel: divided in nine intervals.

-The *Saturation* channel: divided in twelve intervals with many possible saturation scales.

-The *Value* channel: divided in four intervals with many possibles brightscales.

So the characteristics vector for each image has (9*12*4) numbers.

4.1.3 Color Descriptor - Local Color Histogram

The image is divided in five regions, so the color histogram is extracted from each HSV channel from each region. Worth to emphasizing that this method is affected by rotation, but not by scale apply. The hsv channels was divided in appropriate intervals detailed:

-The *Hue* channel: divided in nine intervals.

-The *Saturation* channel: divided in twelve intervals with many possible saturation scales.

-The *Value* channel: divided in four intervals with many possibles brightscales.

So the characteristics vector for each image has 5*(9*12*3) numbers.

4.2. Distance Comparison

The comparison between the images can use three distance metrics:

4.2.1. Chi-Square

$$dChi2(q, d) = \frac{1}{2} \sum_{i=0}^{M} \frac{(q[i] - d[i])^2}{(q[i] - d[i])}$$

4.2.2. Euclidian

$$dEuclidean(q, d) = \sqrt{\sum_{i=0}^{M}(q[i] - d[i])^2}$$

4.2.3. Manhattan

$$dEuclidean(q, d) = \sum_{i=0}^{M}|q[i] - d[i]|^2$$

4.2.4. Normalization

All characteristic vectors were normalized in [0,1] range to represent the percentage of a given color (Color Descriptor) or frequency (Fourier Descriptor) in the previously quantized ranges. In this way, we can use a distance metric to evaluate the similarity of two characteristics vectors.

It was used the *normalize()* function found in OpenCV library to normalize the vectors.

5. Results

It was made several tests outputting the five most similar images to an input image. For simplifying, we used just the chi-square distance on comparison and
we chose to show only the most 6 similar images.

5.1. Global Color Histogram method

The input image was:

We obtain as output:





Another example was with the input:

We obtain the results:

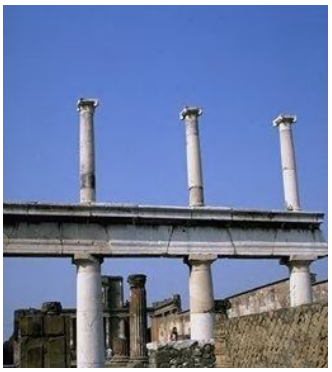
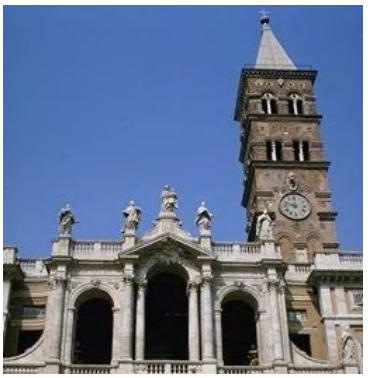





5.2. Local Color Histogram method

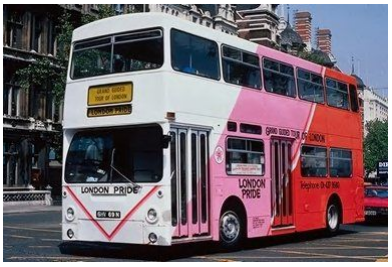
The input image was:

We obtain as output:



Another example of input was:

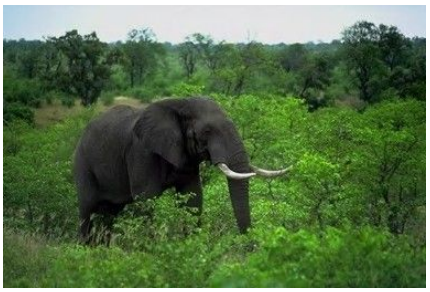


With the follows results:



5.3. Discrete Fourier Transform method

The input image was:



Using the mask 3X3, we obtain the following results:






However, using the mask 7X7, the results were:






Another example of input was:

With the follow result for mask 3x3:






And mask 7X7:






5.4. Results Comparison

The following comparison results is related a test execution using a thousand images to be compared em returning the most fivee similars

5.4.1. Generating Indexes

Method	GCH	LCH	DFT(3X3)	DFT(7X7)
Time	6.05s	11.10s	171.37s	130.17s

5.4.2. Finding similar images

	GCH	**LCH**	**DFT(3x3)**	**DFT(7x7)**
Chi-Square	7.64s	28.52s	45.88s	8.03s
Euclidian	4.24s	15.13s	30.40s	5.01s
Manhattan	3.54s	13.04s	19.52s	4.05s

5.5. Results Discussion

The result discussion considers several tests in a computer, so not everyone was documented in this report.

5.5.1. Timing

Analysing the execution time of the tests done above, we can easily notice that the usage of global color histogram like descriptor is much minor (so it's better), than the others, and that is wasted considering that is a full-image descriptor.

The Discrete Fourier Transform, as a shape descriptor, it was also wasted a major execution time (so it's worst). Using mask 3x3, it takes almost 30 times more than the gch. In addition, as expected, the mask size 3 takes a longer time than that of size 7.

We also noticed in relation to distances, manhattan is more faster than chi-squared and euclidean, but chi-square offer better results in comparison (despite being the slowest).

5.5.2. Performance

When considering the quality results, the performance depends on the image. For being a shape descriptor, Fourier Descriptor can present results very differents from what we expected. In "plate" example, we can notice that the descriptor shows aleatory images like walls and mountains, but also return images of foods that Global Color Histogram or Local Color Histogram could not found, because they wouldn't have a same color similarity. In the

case of the “elephant”, the descriptor presented good results, returning images not only of elephants but also of the nature.

On the other hand, tests with few color variation, the global color histogram shows very good results, like “rose” example, and can even improved using a better distance metric, in case, chi-squared shows the best results. When the query image has great variation of colors, this descriptor can return pictures that don’t have a similarity, like the “indian” example.

Now talking about the the local color histogram, their best results is related on details. Due to extract histogram from specifics regions, the method shows great results on well detailed images as well images with some concentrated color in some specific region. In this cases, even its major time, its usage is much better than the global color histogram.

6. Executing

6.1. Generating index

Considering all the files extracted in a folder, the first step is generate indexes vectors of all the images of the database through some method. Follow the steps:

1- Create a folder inside this workspace folder which will contain all the images (the database). In this case, we already have this folder with the images, called *database*.

2- To execute, follow the command:

python index.py --images <image database directory path> --method <fourier or gch or lch> --mask <3 or 7>.

-The flag --*images* specifies the directory of the images database.
-The flag --*method* specifies the method to be used to compare.
-The flag --*mask* is optional, used in case of Fourier. The default is 7.

Exemple: *python index.py --images database/ --method lch --mask 3*

3- Now there is a *.csv* file inside the *indexes* folder.

6.2. Specifying query

Once generated the indexes, we will execute the desired query, that is, compare an input image to all images of the database. Follow the steps:

1- To execute, follow the command:

python main.py --images <image database directory path> -q <query image path> -n <number of similar images> --method <fourier or gch or lch> --mask <3 or 7> -d <chi2 or manhattan or euclidian>

-The flag --*images* specifies the directory of the images database.
-The flag *-q* or --*query* specifies the image to be compared.
-The flag *-n* or --*limit* specifies how many output images you want.

-The flag --*method* or *-m* specifies the method to be used to compare.
-The flag --*mask* is optional, used in case of Fourier. The default is 7.
-The flag *-d* or *--distance* specifies the distance metric comparison.
Ps: The arguments information can be found using the help command, for example: *python main.py -h*

7. External links

- Repository git: https://github.com/fsampaioo/image_processing

- Example of images: http://wang.ist.psu.edu/docs/related/