

|  |
| --- |
| Middlesex University  Computer Science  CSD3301 |
| MATLAB CW2 – Colour Based Image Retrieval of Flowers |
| Jenny Messer |

|  |
| --- |
|  |

Abstract

Computer systems have a challenging time finding similarity between and categorizing images. A human can easily see if an image is similar to another one, but this takes time. When a large database of images need to be searched and compared a human cannot do the task as it would take too long, so a computer must be used.

In this project comparison of colour histograms of images using the Chi-Square algorithm was used to ascertain how similar the images are overall. An uploaded image was compared to all images in a database and the most similar image was returned to the user.

The project met all aims but could be improved with added features and more user friendly design.

The Chi-Square algorithm comparison of colour histograms was fairly effective as it consistently returns images that are similar in colour composition but not always in shape and content of the image.

Introduction

Image classification comes easily to the human brain, but not so easily to computer systems. There are many methods computer systems can use for image classification including line and edge detection, neural nets and decision tree classifiers. Image classification is used in geographical information systems and remote sensing.

In this project colour histogram comparison is used to ascertain the similarity of images of flowers. Similar images tend to have similar histograms; this means histogram similarity is a good metric for overall similarity.

This report will cover:

* The methodology of the creation of a Colour Based Image Retrieval (CBIR) System
* An evaluation of the project and results of the program created
* A summary of results and a discussion as to how the project could be continued in the future

Methodology

# Aims and objectives

To create a program that can:

* Be developed in MATLAB in version r2017a
* Upload a new image to the system
* Find the colour histogram of an image
* Compare colour histograms
* Find the most similar image from a database, compared to a new uploaded image
* Be easily understood and used by users with a variety of skill levels

# Planning

As this project was undertaken by one person each part of the program was done in succession. The order that the various parts of the program were done in is as follows:

* GUI
* Upload photo capability
* Get photos from database
* Create colour histograms of images
* Research algorithms to use to compare histograms
* Compare histograms capability
* Compare uploaded picture to every image in database
* Show most similar image in GUI

# Resources

I wrote most of the code myself in MATLAB 2017a. The code I did not write consists of code generated by GUIDE, the gui creation tool built in to MATLAB, and the distChiSq function which I accessed from [*http://www.cs.columbia.edu/~mmerler/project/code/pdist2.m*](http://www.cs.columbia.edu/~mmerler/project/code/pdist2.m)

# The algorithm

During my research I found there were a lot of different algorithms for comparing histograms. These include:

* Kullback-Leibler Divergence
* Jenson-Shannon Divergence
* Jeffrey Divergence
* Chi-Square
* Kolmogorov-Smirnov
* (Histogram) Intersection
* (Histogram) Match
* Quadratic form
* Log Likelihood Statistic

There is also the function pdist2 that is already built in to MATLAB. Pdist2 returns a pairwise distance between two sets of observations; this function uses Euclidian distance, which is not very effective for this project.

I chose the Chi-Square algorithm to compare histograms as the authors of LBP method emphasize (Face Description with Local Binary Patterns: Application to Face Recognition. 2004) that Chi-Square distance performs better than Histogram intersection and Log-likelihood statistic.

Evaluation and results

# Results

The program consists of eight MATLAB files: *cw2\_GUI, searchDatabase, chiSq and distChiSq*, *getGlobalDatabaseFolder, setGlobalDatabaseFolder, getGlobalUploadedFile and setGlobalUploadedFile*.

### cw2\_GUI

This program was originally created with GUIDE *(Figure 8)* , MATLAB’s GUI creation tool; It was then modified manually. Most modification were made inside button press events.

The GUI *(Figure 1)* consists of two axes where images can be placed and four buttons: *Select Database Folder, upload image, search database and quit*.

### Buttons:

#### Select database folder

When this button is pressed a pop up file selection window *(Figure 2)* is triggered. The user must then select any file from the folder that contains the database of images. The images must all be .PNG files as the program will ignore any other files.

#### Upload Image

This button triggers a pop up file selection window *(Figure 4)* to appear that allows the user to select an image from their computer to upload to the program. The selected image is then shown in axes1.

#### Search Database

This button gets the database, number of images and uploaded image and sends them to the *searchDatabase* function. The closest matching image is shown in axes2 *(Figure 6)*.

#### Quit

This button exits the program.

### Functions:

#### Search database

This function takes three arguments, *uploadedFile, database, numImages.*

the uploaded file is read into the program along with every file in the database. In a for loop each database image is compared to the uploaded file using the function *ChiSq*. The image with the lowest difference from the uploaded image is returned by the function.

#### ChiSq

This function accepts two image files. The images are made to be the same size and are then split into their RGB channels. the colour channels of picture1 are compared to the colour channels of picture2 using *distChiSq*. This gives us 3 distances. The distances are added together and returned as one total distance between the pictures.

#### The get and set functions

The functions *getGlobalDatabaseFolder, setGlobalDatabaseFolder, getGlobalUploadedFile and setGlobalUploadedFile* are used to get and set the global variables *DatabaseFolder* and *UploadedFile*.

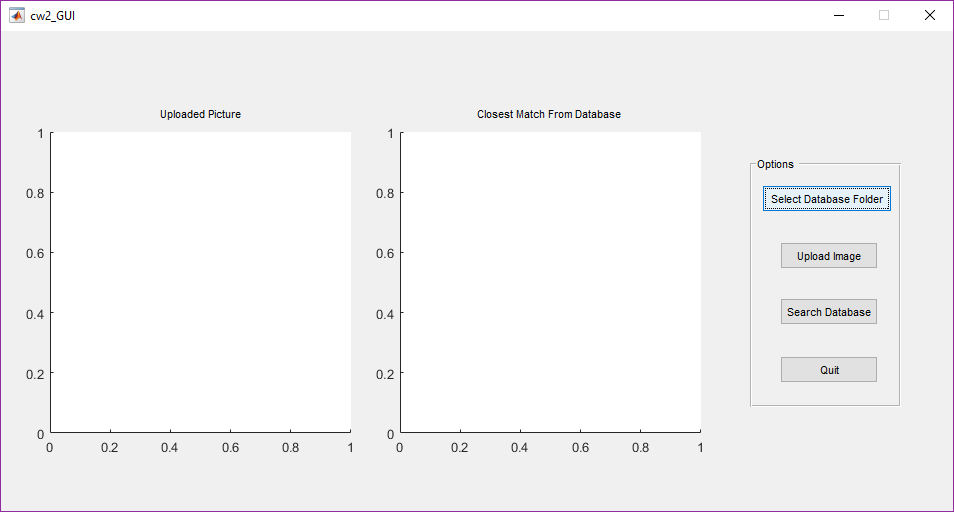


Figure 1 - GUI when first run

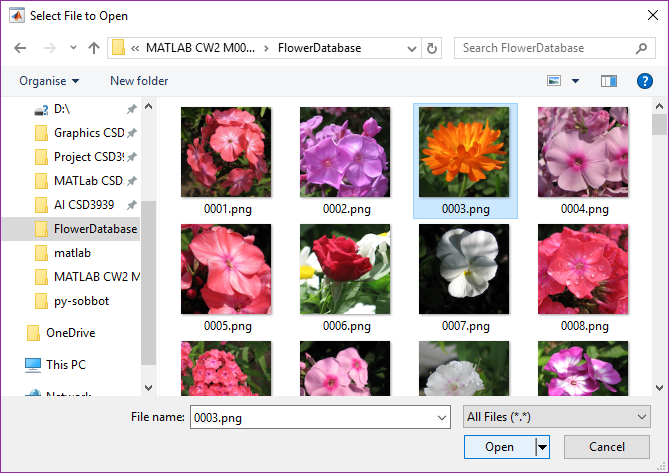


Figure 2 - pop up folder selection window

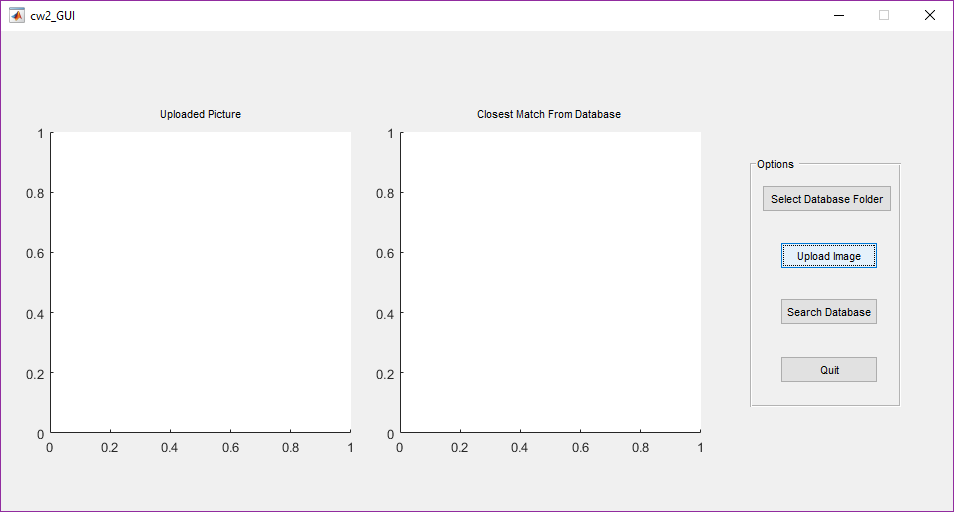


Figure 3- GUI after folder selection

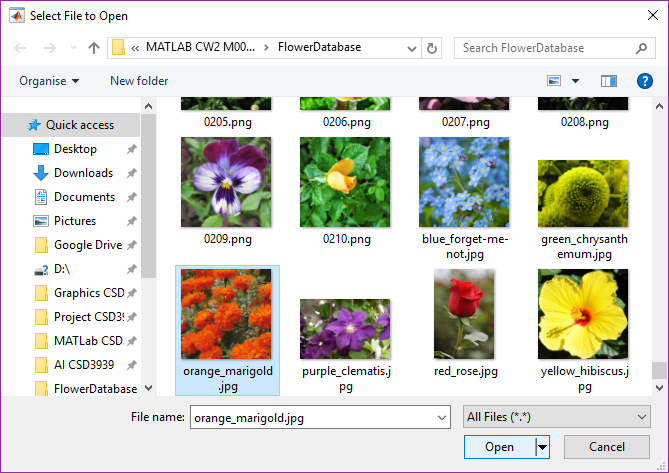


Figure 4 - a pop up file selection window

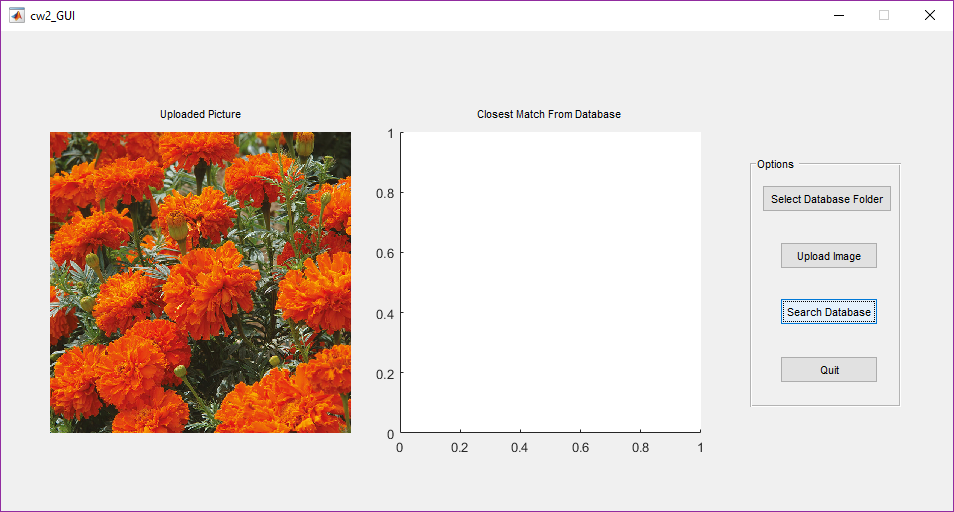


Figure 5 - GUI after file uploaded

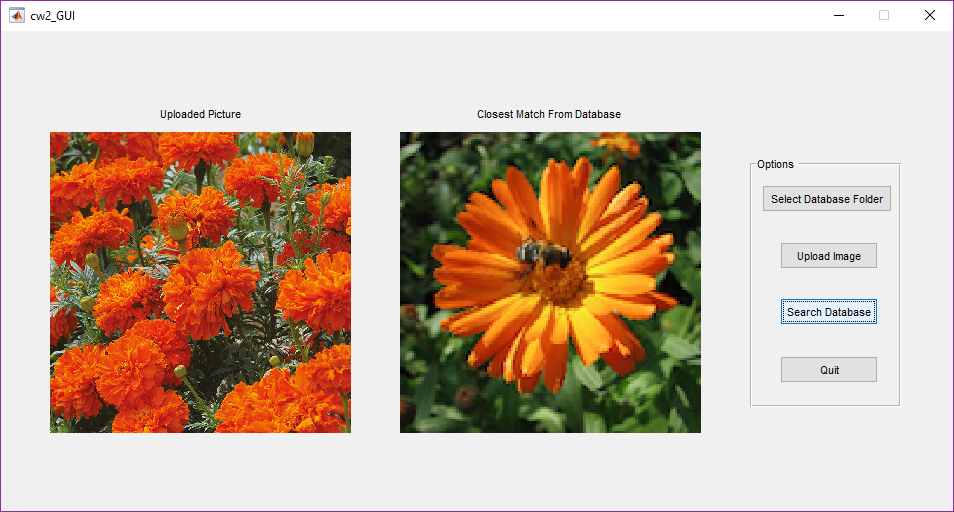


Figure 6 - GUI after 'search database' button pressed

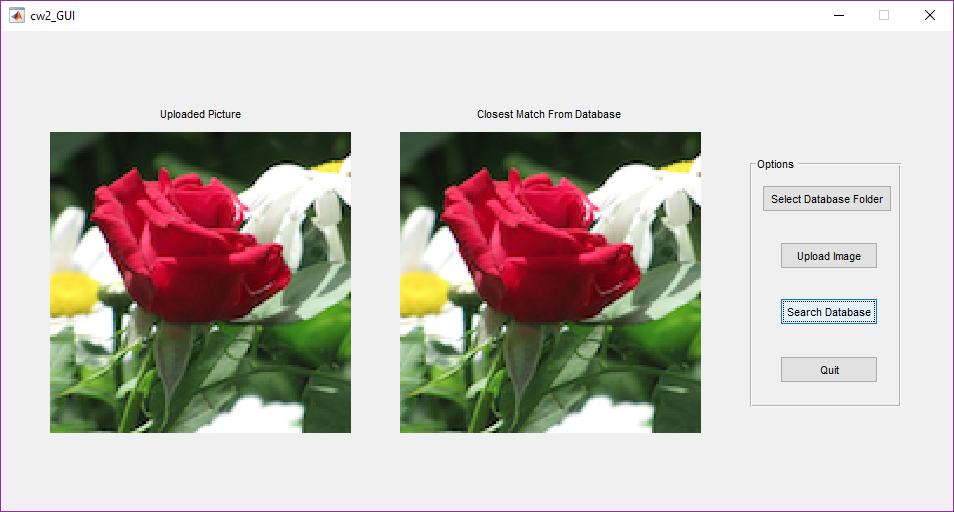


Figure 7 - GUI after database searched for image it already contains

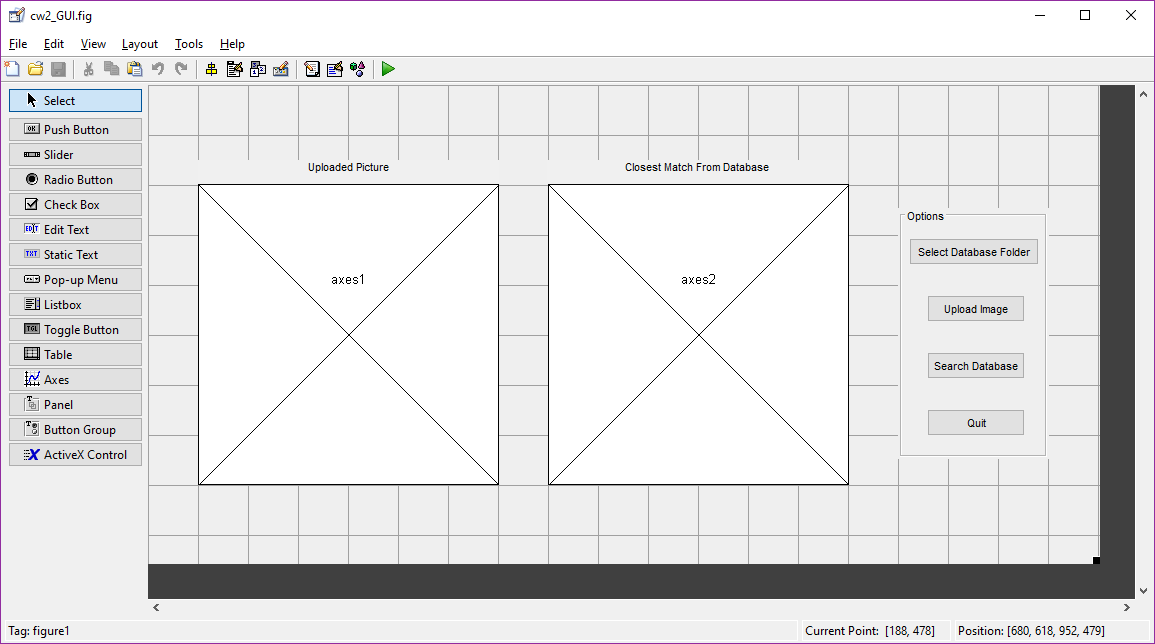


Figure 8 – cw2 GUI in GUIDE creator

# Evaluation

The project was successfully implemented in MATLAB r2017a. The program can upload a new image to the system, create colour histograms, compare histograms and find the image in the database most similar to the uploaded image. The program is easy to use as the GUI is clear and simple to interact with.

It is hard to know how well the program finds the most similar image though, as there are many similarity metrics and I have not been able to test them all.

Summary and discussion

# How the project could be improved

The program currently only searches the database for .PNG images, this is an unnecessary constraint and could be easily changed to make the database easier to manage. It would be easier to manage with all image file types accepted as then the database manager could add any file to the database, whereas currently they would have to convert the image to a .PNG if it is not already a .PNG. The program only recognizes .PNG images to avoid non-image files that may be in the folder being read into the program, which would cause an error.

When the user wants to select the database folder, they have to select a file that currently resides in that folder. It would be easier for the user to simply select the folder itself. The decision to make the user choose a file in the needed folder instead of the folder was made because the code to access a file from a folder was already in use in the program and it was easy to use that code to select a folder, instead of attempting a different method.

The GUI could have been more aesthetically pleasing. The GUI was created with GUIDE and therefore looks very generic. This means the GUI is easy to use as most people are used to using a GUI that looks like this, but it is also boring and un-professional. It is un-professional as more programs created with larger budgets by large companies have more stylized GUIs; a better looking GUI is a signifier of a good program.

# Possible future work

A good feature for the program to have in the future would be to show all images in the database in order of similarity to the uploaded photo. This would give the user a higher chance of finding a similar photo.

The histograms of the photos could be shown next to the images. This would allow the user to see how similar the histograms look visually.

Showing the names of the files next to the images would be helpful for the user as they may wish to find the image file that is most similar to the uploaded image; without a file name they would have to manually search the database to find the image and this could be so time consuming as to be impossible.

Currently the database only has 210 images, but a larger database would make the program more likely to find a good match to the uploaded image.

An additional algorithm could be added to the program that detects lines, this would help differentiate between pictures of many flowers and pictures that are a close up of one flower.

# Chi-Square algorithm

The algorithm chosen to compare the histograms is what determines how well this program found the most similar image.

When the program was asked to find the most similar image to an image already in the database, it returned the identical image. This means it can identify when two images are at maximum similarity and return the correct picture.

When a new image (not in the database) is uploaded to the program, the program returns an image that, to the human eye, looks very similar. The same type of flower is not always returned however; for example a sunflower will not always return another sunflower, the program may return a daisy as a daisy has a similar colour histogram. Also an image of one large flower could have a very similar histogram to an image of many small flowers, which are very different to the human eye.

# Conclusion

The method of finding which images are similar using Chi-Square histogram comparison is fairly effective, but not yes as effective as if the human eye were to classify images. The program is useful but not accurate enough to be used to reliably guess what type of flower a picture is of.