Jeremy Woods  
CSS 590  
Assignment 1

**How to run program**

This program has 2 executable parts. One being readImage.exe and the other being CBIR.exe. I have created a batch file that has the ability to run both of these with one click. If you go into the Assignment one folder and click on launcher.bat both parts of the program will run and that is all that is need. If you do not wish to do it that way, first launch readImage.exe and then run CBIR.exe. It is done in this order so that readImage.exe can first create the files that CBIR.exe will need.

**Step-by-step use of System**

When the system first comes up there will be an empty panel to the upper left, six buttons to the upper right, and at most 20 pictures on the bottom panel. Here you must either select a picture from the bottom panel or page through the pictures with the next and previous buttons. Pressing any of the query buttons will cause a message to appear in the upper left panel prompting the user to pick a picture for the query. If you select the “Return Pictures to Original Order” button nothing will happen because nothing has been queried yet and as a result the pictures are still in their original order.

After a picture is selected, it will appear in the upper left panel with its name under it. Now you can press any of the query buttons in order to sort all the pictures based on that query. If you select the “Query Images by Intensity” button, the pictures will be sorted in the order or pictures closest to the intensity of the selected picture. If you select the “Query Images by ColorCode” button, the pictures will be sorted in the order or pictures closest to the colorcode of the selected picture. If you select the “Query Images by Intensity + ColorCode” button, the pictures will be sorted in the order or pictures closest to the sum of the Intensity and ColorCode of the selected picture.

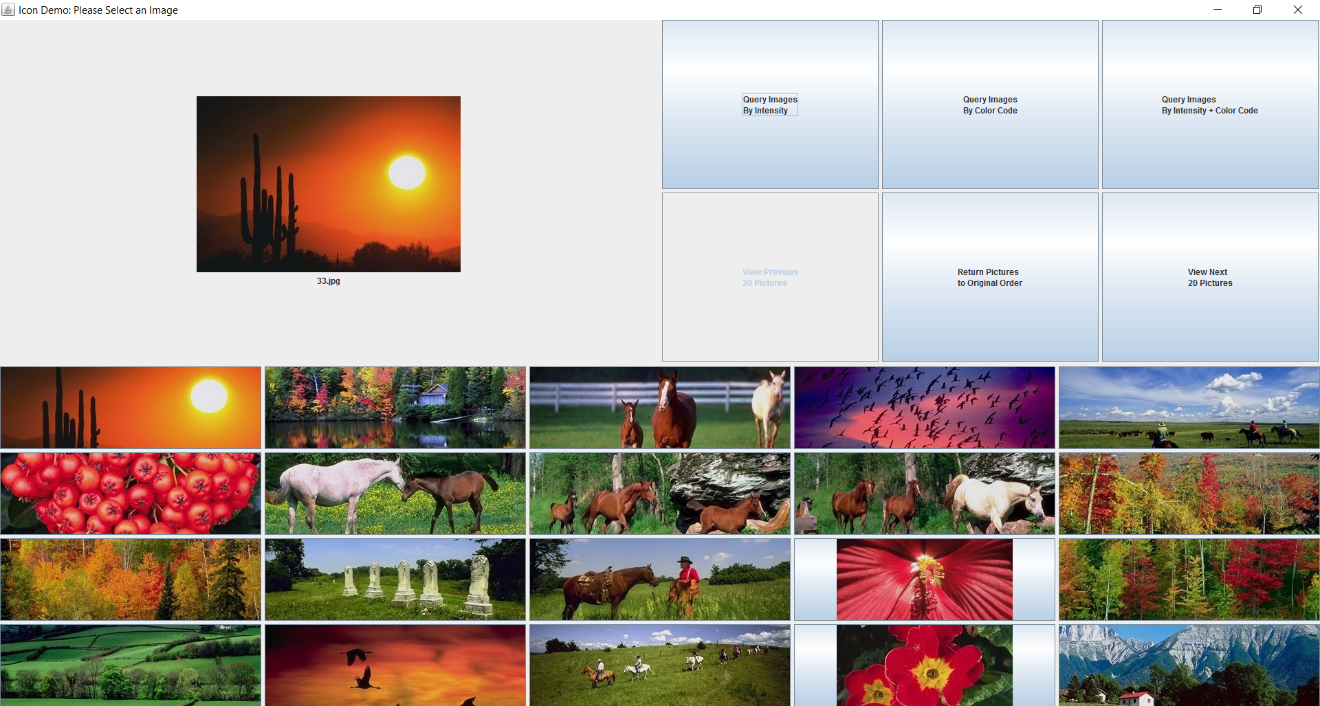
After a query has been done, the closest 20 pictures will be shown in the bottom panel, and as the next button is pressed the list will descend to the pictures the most different query attribute. Also, the “Return Pictures to Original Order” button can be pressed at any time to have the pictures in the bottom panel go back to the order they were in when the application was first started.

**Code Used**

I used the GUI code provided by the professor for this implementation. I changed some things to improve user friendliness and also I changed some of the methods to be better reused throughout the program.

**Screen Dump**

*Figure 1*: Pic 33 Intensity



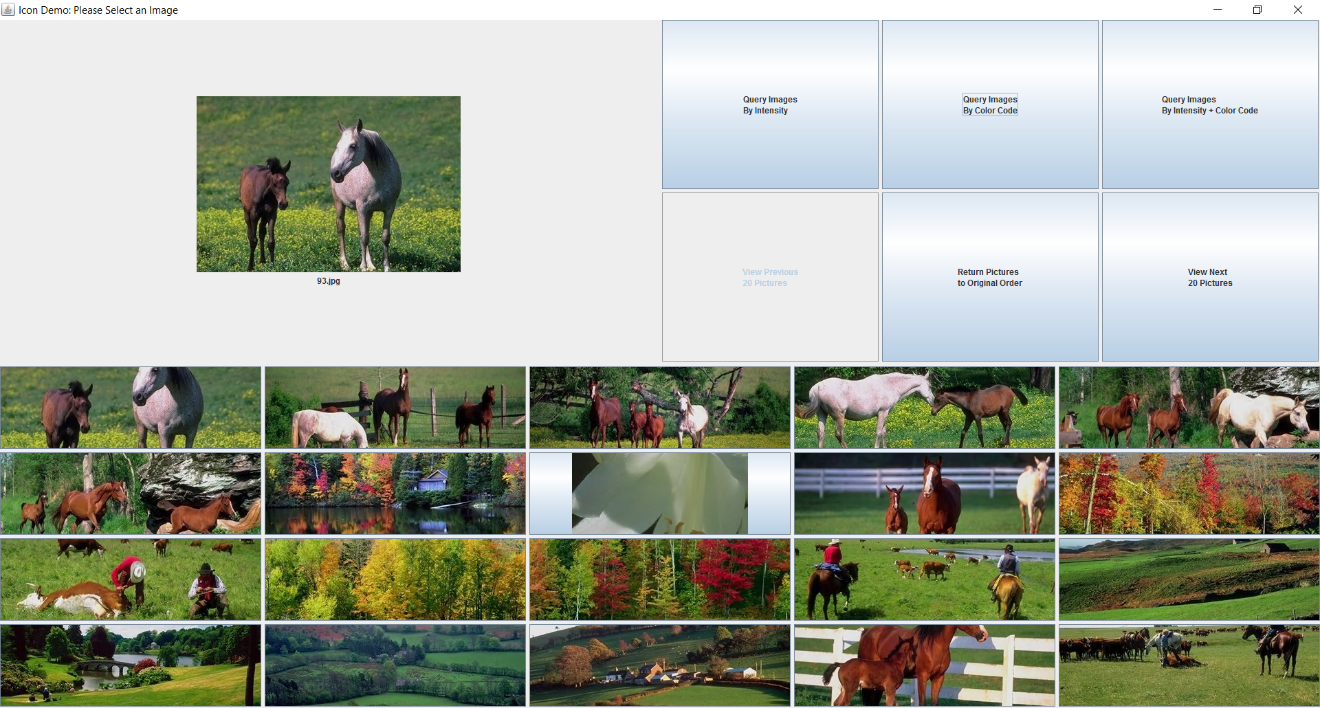
*Figure 2*: Pic 33 ColorCode



*Figure 3*: Pic 93 Intensity



*Figure 4*: Pic 93 ColorCode



**Further Analysis**

*Advantages*

Designer

This implementation is very straight forward and simple. Using the heuristic bins in order to compare images gives a simplistic baseline for finding the differences between two images. Both the intensity and colorCode implementations were very similar, so have getting one correct it was very easy to apply the same code to the other, which made the code easily modularized.

User

The system is easy to use, from launching it, to the ordinary use. The feature that blends both of the algorithms together makes for better analysis between images. The feedback from the program such as the no picture to query message and the greyed out buttons help to understand what is next and eliminate error. The added name on the picture that is selected helps to identify what I am querying. Returning the pictures back to their natural order help with next selections.

*Limitations*

Designer

Requires that files be written and read from. The bins in both matrices are not aligned exactly the same and that makes for some extra work and increased coupling to make methods reusable. Difficulty dealing with JPanel in java.

User

No names under each picture so it is difficult to know what picture you are selecting before you choose it. The comparisons between the pictures are rather juvenile, meaning that the results could be better.

*Overcoming Limitations*

Designer

We could move it to a database. Getting the code from a database could be faster than having to write to a file every time another picture should be added to the options. If we were to move this to a database, we could also hold each picture’s size in a column and therefore all of the matrices can be started from 0 and we would still have access to the picture sizes without having to recalculate each time. The reason that the names are not next to the pictures is because when dealing with JButtons, adding text in the small frame of the picture would cause the images to be altered. The final decision was the forgo the names of the pictures in order to keep the resolution of the pictures the same. With more time, I would implement a boarder around the pictures, as to maintain their quality while allowing space for the names to be displayed at the bottom of the button.

User

Moving from RGB to HSV could help to get the images closer. Also exploring other image comparison algorithms. Algorithms such as SIFT (Scale-Invariant Feature Transform), and SURF (Speeded Up Robust Features) could be helpful to maybe get our results closer instead of just heuristics, blended with the Manhattan distance formula.

SIFT consists of these 5 steps: Scale-space Extrema Detection, Keypoint Localization, Orientation Assignment, Keypoint Descriptor, Keypoint Matching.

SURF is a speeded up version of SIFT.

***Resources***

Docs.opencv.org  
Wikipedia.org  
http://www.javatpoint.com/  
http://stackoverflow.com/  
https://docs.oracle.com/