Team - Ronak Bhanushali and Prem Sukhadwala

Project Description

This project explores concepts of Content Based Image Retrieval (CBIR) in the field of Computer Vision. There are multiple CBIR methods and we have implemented a few like – Baseline Matching, Histogram Matching, Multi-Histogram Matching, Texture and Color Matching, and DNN Embedding.

Each task delved into different methods. The project comprises seven tasks and an extension part.

(Order of the images are in ascending order from left to right in each case)

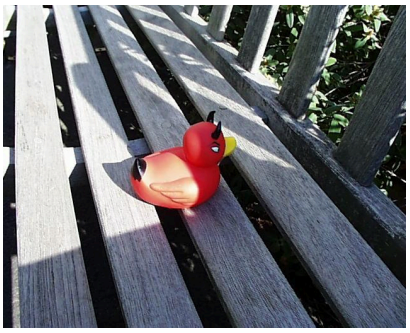
Task 1: Baseline Matching

We took the second implementation for this task, i.e. making it a two part problem and using a CSV file to reduce the time complexity.
The top three matches (ascending order) for the target image pic.1016.jpg are **pic.0986.jpg**, **pic.0641.jpg**, **pic.0547.jpg**.

Target Image (pic.1016.jpg)

Top matches

pic.0986.jpg



pic.0641.jpg

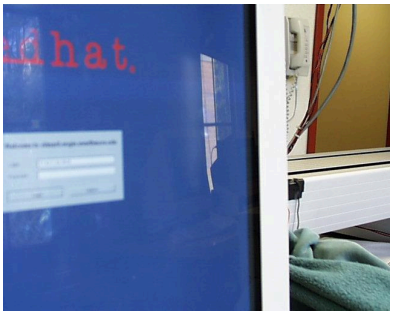
pic.0547.jpg

Task 2: Histogram Matching

For Histogram matching, we used an RG histogram which gives us chromaticity values. The bin size used for this task was 32 and we got the following matches-

Target Image (pic.0164.jpg)

Top three matches



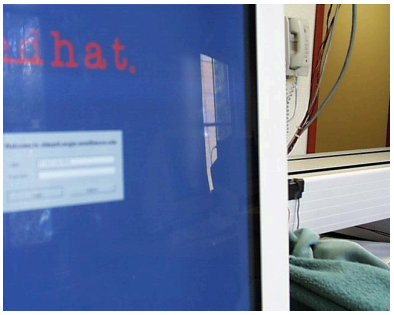
pic.0080.jpg

pic.0110.jpg

pic.0116.jpg

Extension 1

For the first extension, we tried implementing HS histogram instead of RG histogram for matching. We got the following three images as matches -



pic.0080.jpg

pic.0898.jpg

pic.0110.jpg

All three have a lot of blue pixels like the original image.

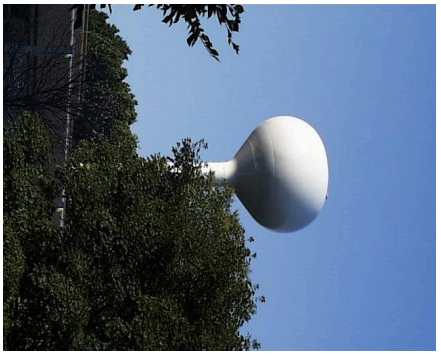
Task 3: Multi-histogram Matching

For this task, we used the HS histograms (bin size 32) for the whole image and matched the HS histogram (bin size 32) of an area of 150x150 from the center of the image. Combination of these two gave the following result -

Target Image (pic.0274.jpg)

pic.0825.jpg

pic.1055.jpg



pic.0209.jpg

Task 4: Texture and Color

For this we used the magnitude of sobel X and sobel Y to get the edges. Then we made an RG histogram of it to use as one comparison metric. Another comparison metric was just the RG histogram of the image. These are the results -

Target Image pic.0535.jpg

pic.0004.jpg

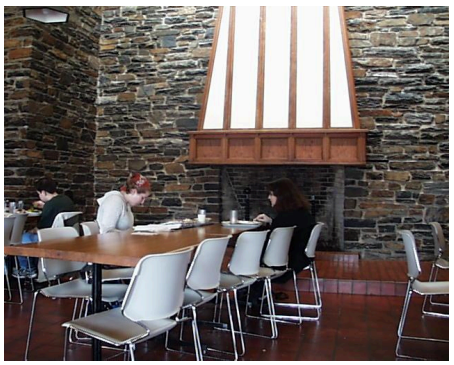
pic.0011.jpg

pic.0628.jpg

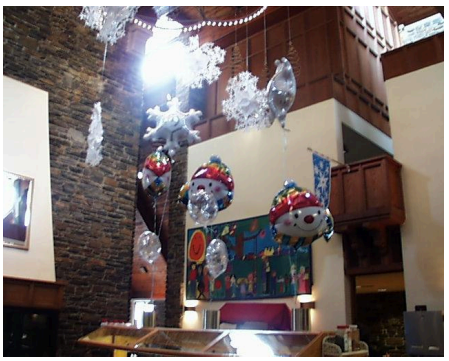
Extension 2: Gabor Filters-

For this extension, we used a gabor filter in place of sobel to get the features. We then took an RG histogram of it and the rest of the comparison is the same as above. The metric used in both is intersection area.

Target Image pic.0535.jpg



Top 3 Matches-


pic.0628.jpg


pic.0731.jpg


pic.0628.jpg

Task 5: Deep Network Embeddings
Distance metric used for this task is the L2 norm.
The top three results for pic.0893.jpg are **pic.0897.jpg**, **pic.0136.jpg** and **pic.0885.jpg** and the top three results for pic.0164.jpg are **pic.0213.jpg**, **pic.1032.jpg** and **pic.0543.jpg**

Target Image (pic.0893.jpg)

Top matches

pic.0897.jpg

pic.0136.jpg

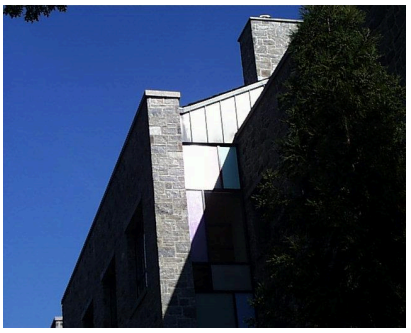
pic.0885.jpg

Target Image (pic.0164.jpg)

Top matches

pic.0213.jpg

pic.1032.jpg



pic.0543.jpg

The result for target image pic.0164.jpg with DNN embeddings is quite different from what we got with histogram features in Task 2. In Task 2 with color histogram, the program selects images based on how blue the image is. As a result, we can see that the top matches consists of blue pixels but does not necessarily have a sky and a gray building. On the contrary, the matches we get from DNN embeddings all have a sky and a gray building. This can be attributed to the DNN feature vector containing information about multiple aspects like color, texture, edges, shapes depending on how the intermediate layers have been trained. Since the embedding has a lot of information represented in each element, the embeddings of two images with a gray stone building and sky will be quite similar and that is what we see in the results too.

Task 6: Compare DNN Embeddings and Classic Features

Upon comparing DNN embeddings and classic features we found that DNN outperforms all the classic features for the case of all the following three images – pic.1072.jpg and pic.0948.jpg. On the other hand just baseline color matching performs the worst which is obvious. And depending on the type of image multi-histogram performs better than single histogram features.

To keep the report length short we would not showcase all the images for each and every case but just give the image number.

Target Image (pic.1072.jpg)

Top DNN matches

pic.0143.jpg

pic.0329.jpg

pic.0940.jpg

Top color baseline matches

pic.0138.jpg

pic.0303.jpg

pic.0768.jpg

Top RG histogram matches

pic.0142.jpg

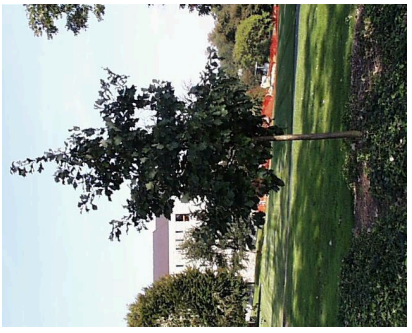
pic.0937.jpg

pic.0940.jpg

Top multi-histogram matches

pic.0673.jpg

pic.0859.jpg



pic.0866.jpg

Top texture-color matches

pic.0937.jpg

pic.0233.jpg

pic.0936.jpg

From the above images it can be seen the DNN performs the best and color baseline performs the worst. Notably, RG histogram and texture-color performs better than multi-histogram for this target image. But if we check the results for pic.0948.jpg, multi-histogram performs better than both RG histogram and texture-color.

Target Image (pic.0948.jpg)

Top multi-histogram matches

pic.0513.jpg

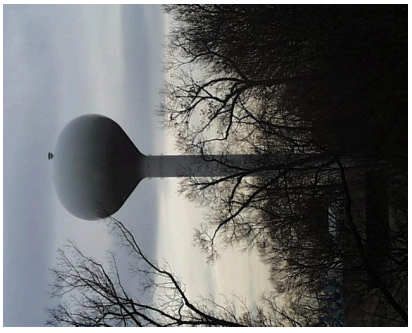
pic.0930.jpg

pic.0950.jpg

Top RG histogram matches

pic.0460.jpg

pic.0619.jpg



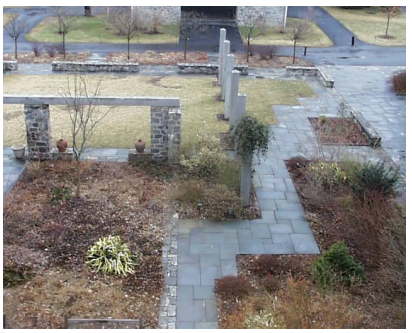
pic.0727.jpg

Top feature-color matches

pic.0727.jpg

pic.0513.jpg

pic.0574.jpg



From what we have observed, the reason for such a performance is that when there is a clear object of interest in the center of the image (ex. pic.0948.jpg), multi-histogram performs better. And when the color/texture data is more spread out in the image, single histogram (RG or HS) and texture-color based matching performs better than multi-histogram.

Task 7: Custom Design

For this task we divide all query images into four quadrants and a fifth center rectangle. The dimensions of all the rectangles are the same and calculated based on the image size. We take a central rectangular ROI of the same size from the target image and match it with all the five rectangular ROIs of the query images using RG histogram and histogram intersection. The ROI with the highest match will be used to perform texture matching and the result from both matches will be combined by a weighted sum. In our method we have given texture a weightage of 0.3 and RG histogram a weightage of 0.7. Following are the results -

Target Image (pic.0287.jpg)

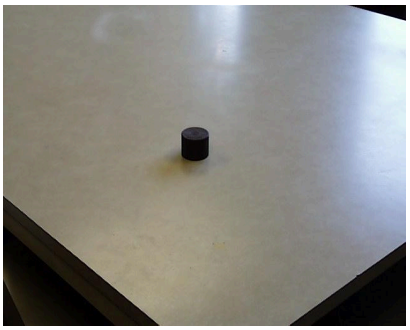
Top matches

pic.0289.jpg

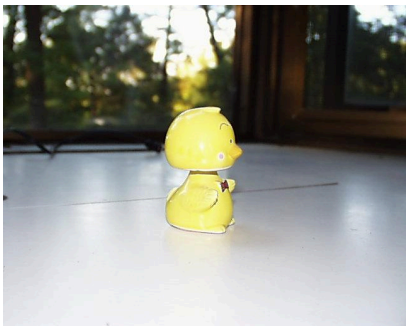
pic.0203.jpg

pic.0291.jpg

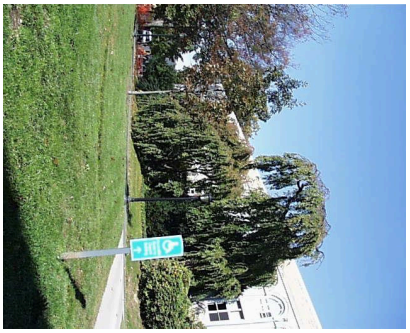
pic.0400.jpg



pic.0469.jpg



Target Image (pic.0997.jpg)

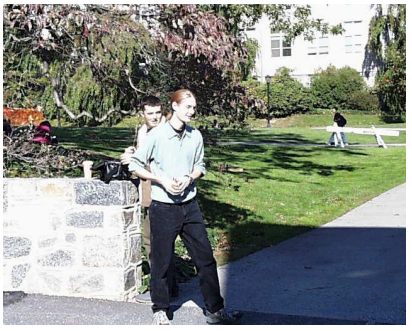


Top matches

pic.0711.jpg

pic.0094.jpg

pic.1033.jpg



pic.0234.jpg

pic.0807.jpg

Reflections:

- With this project we learnt how to collaborate using source-control
- Followed C++ standards for directories and comments
- Learned how to use CSV files for improving computation times
- Realized how different features can affect different images for CBIR
- Gabor Filters working and visualization on images
- How to use different histograms for CBIR
- Learned how to choose different methods for different images (for eg: which images are best for texture)

Acknowledgements:

- Code from Professor Maxwell to create RG histograms
- OpenCV Documentation
- Stack Overflow for debugging
- CMake.org for help with CMake