Ruohe Zhou and Rucha Pendharkar 9th February 2024

Project II
Content based Image Retrieval

I. Description

This project is to get familiarized with C++, the OpenCV package, and the mechanics of opening, capturing, manipulating images at the pixel level. It also involved implementing matching, or pattern recognition dealing with textures and Histograms.

The scripts were authored using VS Code, and code compilation took place in the Ubuntu 20.04.06 LTS environment, utilizing CMake through the terminal

For this project the inputs to the system were a target image, an image database, a method of computing features for an image, a distance metric for comparing the image features from two images, and the desired number of output images.
The output will be an ordered list of the images in the database that are most similar to the target image, according to the distance metric and features, in ascending order of distance

The following tasks have been defined:

1. Baseline Matching
2. Histogram Matching
3. Multi Histogram Matching
4. Texture and Color
5. Deep Network Embeddings
6. Compare DNN Embeddings and Classic Features
7. Custom Design
8. Extensions

II. Tasks

1. Baseline Matching

Top 3 results for the two target images -

Target Image - pic.1016

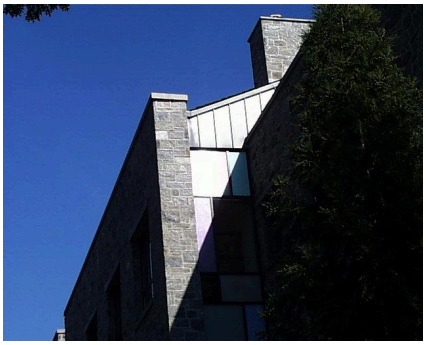
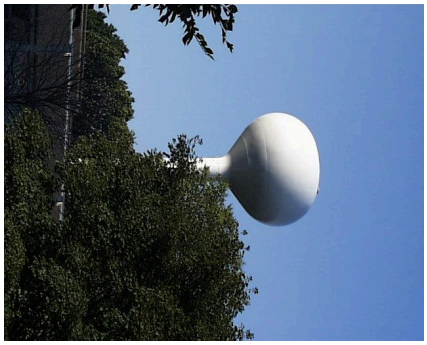
1st Match - pic.0986

2nd Match - pic.0614

3rd Match - pic.0547

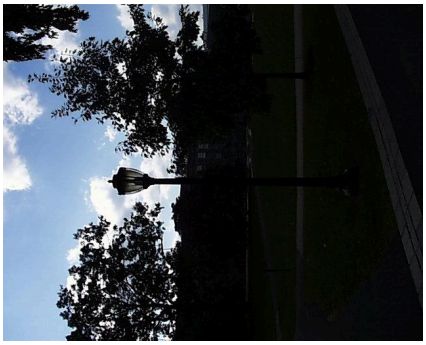
2. Histogram Matching

Target Image - pic.0164

		
1st Match - pic.1032	2nd Match - pic.0209	3rd Match- pic.1055

3. Multi Histogram Matching

Target Image - pic.0274

		
1st Match - pic.0409	2nd Match - pic.0273	3rd Match- pic.0822

4. Texture and Color

Target Image - pic.0535

1st Match - pic.0150

2nd Match - pic.0731

3rd Match- pic.1105

Compared to the outcomes of tasks 2 and 3, task 4, which integrates the whole image color histogram with the whole image texture histogram, can more effectively capture both the overall color distribution and the texture characteristics of the image, surpassing the approach of using two or more color histograms separately.

5. Deep Network Embeddings

For this task, Cosine Distance was chosen as the distance metric.
Top 3 results for the two target images -

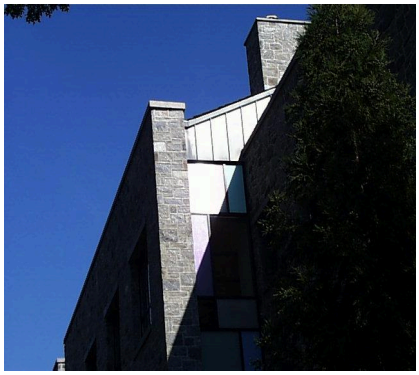
Target Image - pic.0893

1st Match - pic.0897

2nd Match - pic.0136

3rd Match - pic.0146

Target Image - pic.0164



1st Match - pic.1032

2nd Match - pic.0213

3rd Match - pic.0690

6. Compare DNN Embeddings and Classic Features

Overall - it can be found that the DNN Embedded feature vectors perform really well and can effectively capture all the various characteristics. Sometimes however, I observed the color is not captured effectively. But they are still better than all the other vectors. The following are the comparative results - the top row denotes the target image, the middle row denotes the matches from the DNN Embedded vectors and the bottom row denotes matches using features generated in task 1

Target Image - pic.1072

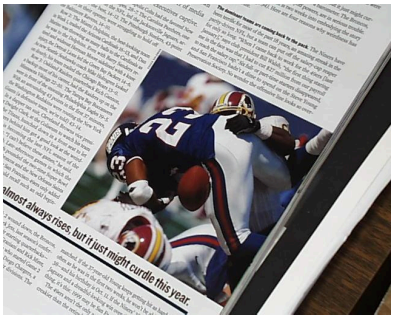
		
1st Match - pic.0143	2nd Match - pic.0863	3rd Match- pic.0329
		
1st Match-.0937.jpg	2nd Match - pic.0863.jpg	3rd Match- pic.0431.jpg

Target Image - pic.0948

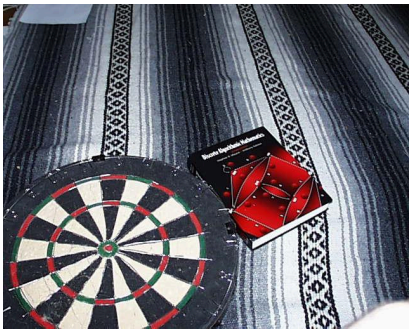
1st Match - pic.0930

2nd Match - pic.0960

3rd Match - pic.0928



1st Match - pic.0064



2nd Match - pic.0176

3rd Match - pic.0176

Target Image - pic.0734

1st Match - pic.0731

2nd Match - pic.0735

3rd Match - pic.0739

7. Custom Design

This task focuses on detecting the red furry stuffed toy in the image dataset. A smaller dataset consisting of images pic.0930 to pic.0990 were used for this task for designing.

The ResNet features are really good at capturing high level features and overall pattern matching. While the feature vector does contain some information about color, it may not be the best for performing color based image retrieval. To remedy this we used - ResNet18 feature vectors and RGB chromaticity histogram. The red channel is used primarily for detecting the red color. Another set of features were extracted from the center of the image, used to compute the histogram distances (from Task 2)

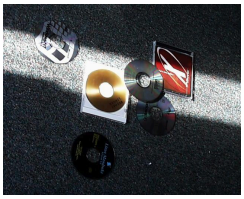
A custom distance metric was defined which is the weighted average of the matches obtained from both the ResNet and the outputs from the histogram. Weights of 0.7, 0.3 were assigned to the feature and color features, which yield the best results (70% accuracy for the dataset)

Two target images containing the stuffed toy were tested out and the following results were obtained -

pic.0928

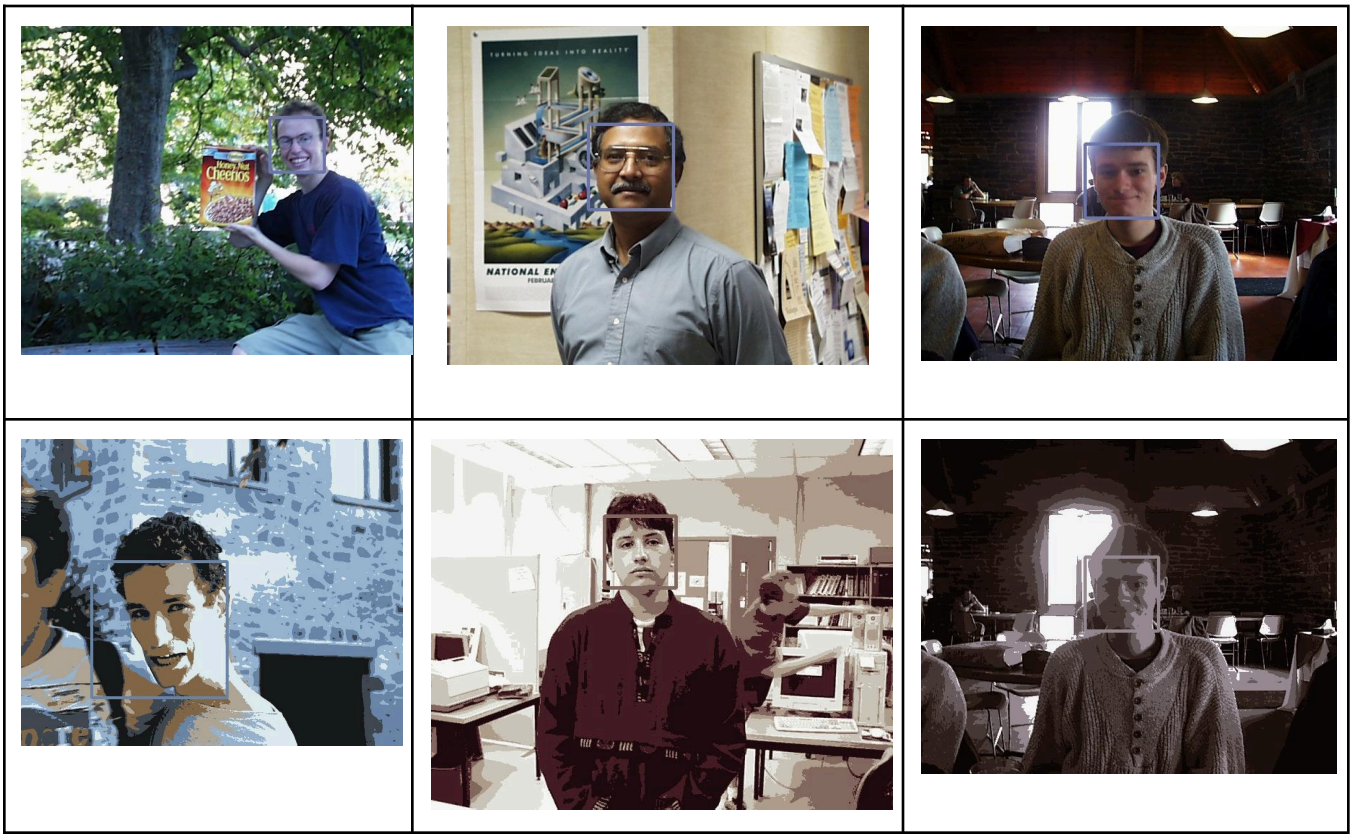
				
pic.0960	pic.0930	pic.0948	pic.0937	pic.0367

pic.0930

				
pic.0948	pic.0960	pic.0367	pic.0930	pic.0467

8. Extension - Human to Cartoon Convertor

The extension goes through the picture database and returns images with the human face detected and returns the images with the face highlighted with a bounding box. Then we perform cartoonization using Kmeans on the images with detected faces. Here we essentially use “yes / no” as a feature vector. Below are three sample of faces detected in the database:



There are also a few cases where it incorrectly detected the face and did cartoonization or failed to detect multiple faces. Overall it gave fairly accurate results.

III. Reflection

In this assignment, we learned about content-based image retrieval and explored various methods for matching images based on their features. We also gained insights of the strengths and weaknesses of different feature extraction methods and distance metrics as well as the trade-offs between computational efficiency and retrieval accuracy.

IV. Acknowledgement

We would like to thank Prof Bruce Maxwell for helping me grasp these new concepts quickly, and for providing various course materials that were useful in completing this project. We would like to thank the Teaching Assistants for their timely help in debugging small errors in my code during implementation of these tasks.

Materials referred -

a. CS 5330 Lecture notes and materials
b. OpenCV documentation
c. StackOverflow