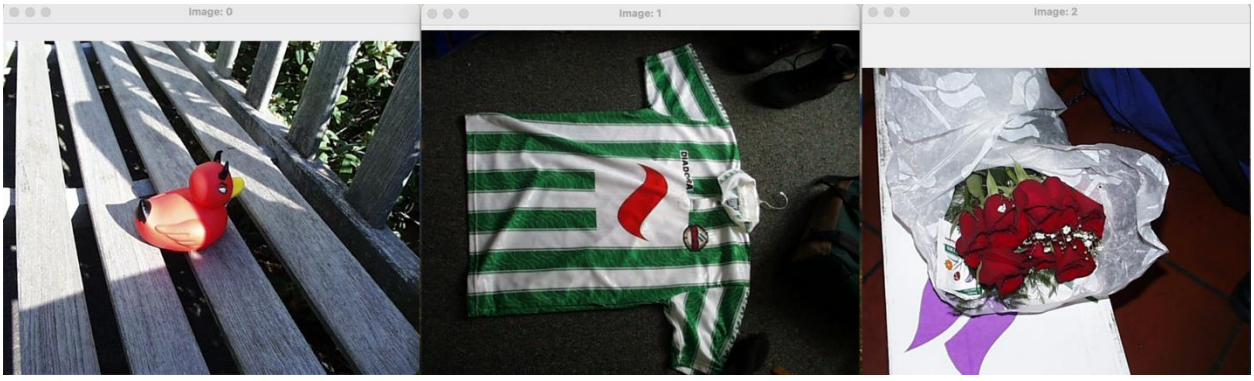
CS 5330 – Project 2: Content Based Image Retrieval

Description

The objective of this project is to delve into the intricate processes of image feature extraction and the various methodologies utilized for image matching. We have developed a comprehensive pipeline function that provides us with the flexibility to experiment with a diverse array of features for extraction, choose from different distance metrics for comparison, and specify the number of top or bottom results we wish to analyze. Furthermore, we have engineered a function capable of transforming a database of images into feature vectors, which are then systematically stored in a CSV file. This innovation significantly accelerates the matching process after the initial execution, enhancing efficiency and facilitating a swift comparison across a vast dataset of images. Through this project, we aim to not only expand our understanding of image processing techniques but also to establish a robust framework for rapid and accurate image matching.

1. Baseline Matching

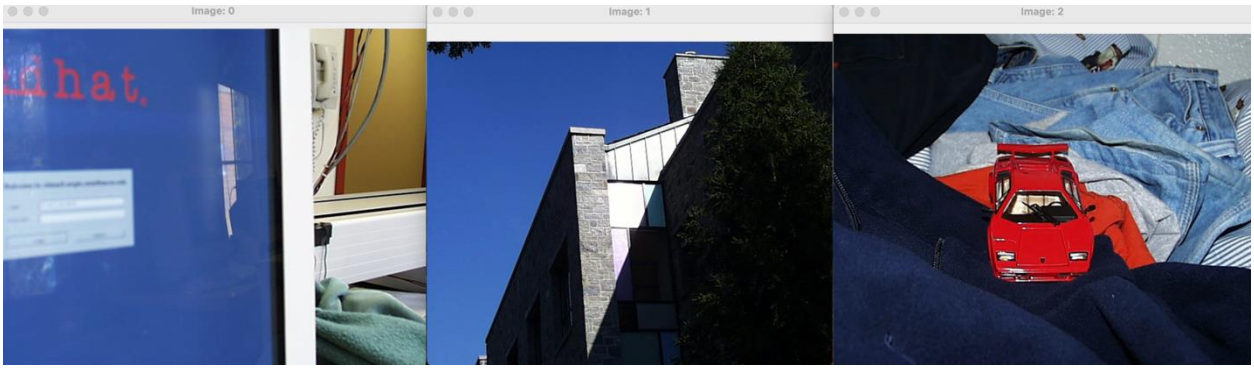
In our baseline matching approach, we focused on a central 7x7 region of interest (ROI) within each image, utilizing the raw pixel values from this area to construct a feature vector. The comparison between images was conducted through the Sum of Squared Differences (SSD) method, a straightforward yet effective technique for assessing similarity between feature vectors. The top 3 images for the target image pic.1016.jpg are as follows:



2. Histogram Matching

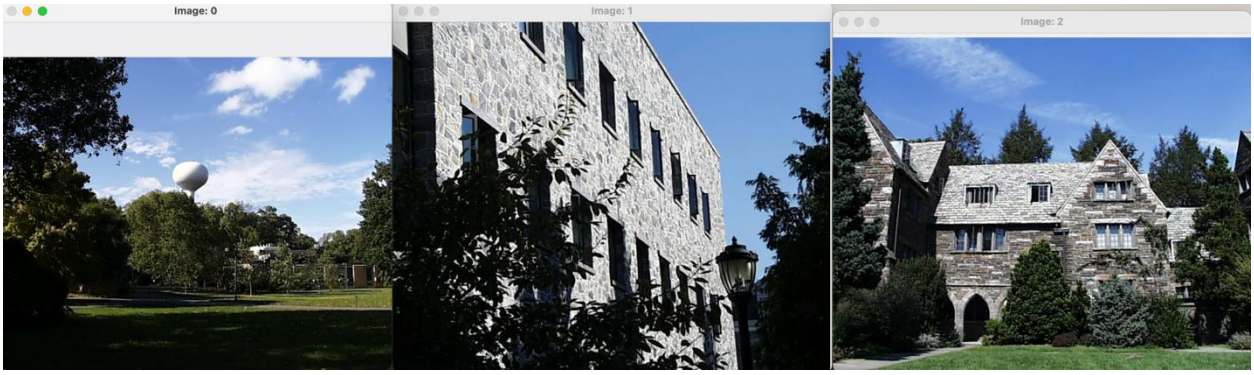
We implemented an RG chromatography histogram as part of our image analysis and matching framework which offers a nuanced approach to understanding image similarity beyond simple pixel values. By focusing on the red (R) and green (G) color channels and quantifying their distributions into 8 bins, we gain insights into the color composition and patterns within images. This method leverages the histogram intersection technique for matching, which provides a

robust metric for comparing the histograms of two images. The top 3 images for the the target image pic.0164.jpg are as follows:



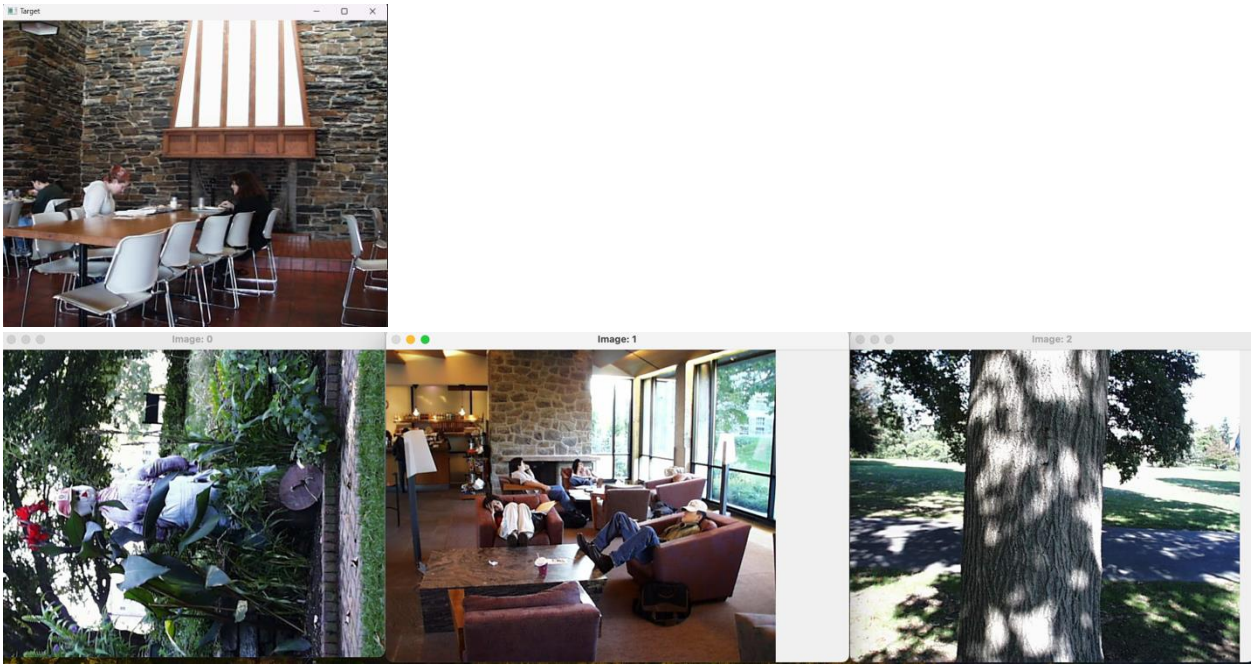
3. Multi-histogram Matching

Dividing an image into two halves and analyzing the RGB histograms for each half separately offers a nuanced approach to understanding the distribution of color across different sections of an image. This method can reveal variations in color distribution that might not be apparent when considering the image as a whole. By averaging the histogram intersection distances of the respective halves, you gain insights into the overall similarity between images, taking into account potential differences in color distribution across their spatial extents. The top 3 images for target image target image pic.0274.jpg are as follows:

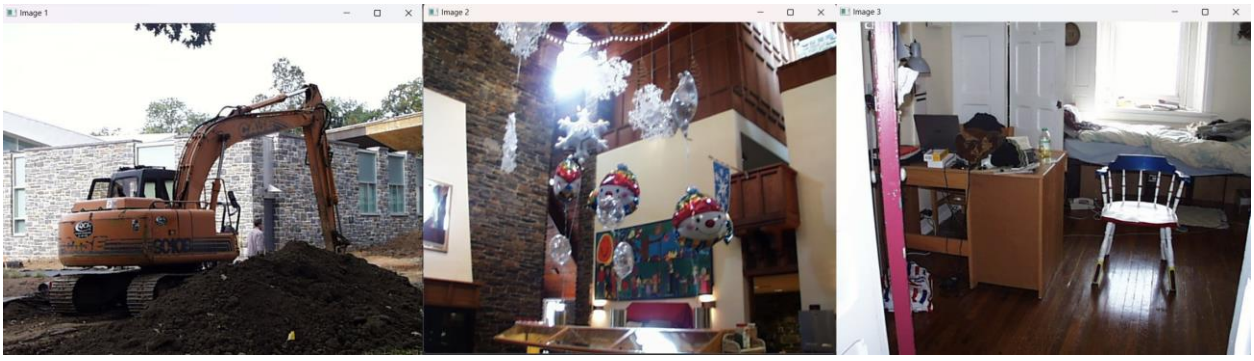


4. Texture and Color

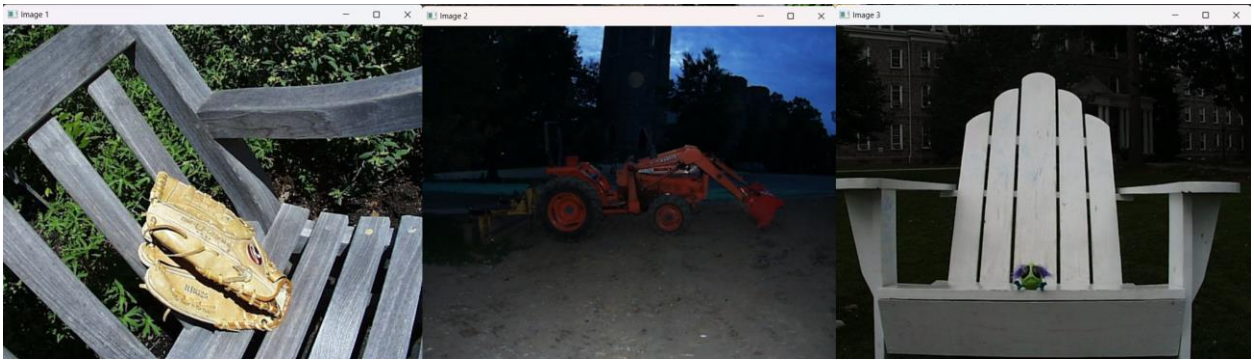
Integrating both texture and color features to analyze images provides a comprehensive understanding of their visual content, enhancing the ability to accurately match or compare images. This approach combines the analysis of an image's texture, using a Sobel magnitude histogram to capture edge information, with the color distribution, using an RGB histogram. The Bhattacharyya distance is employed to quantify the similarity between the histograms, offering a robust measure for comparison. The best matches for target image pic.0535.jpg are shown below



These are the best results using RG histogram



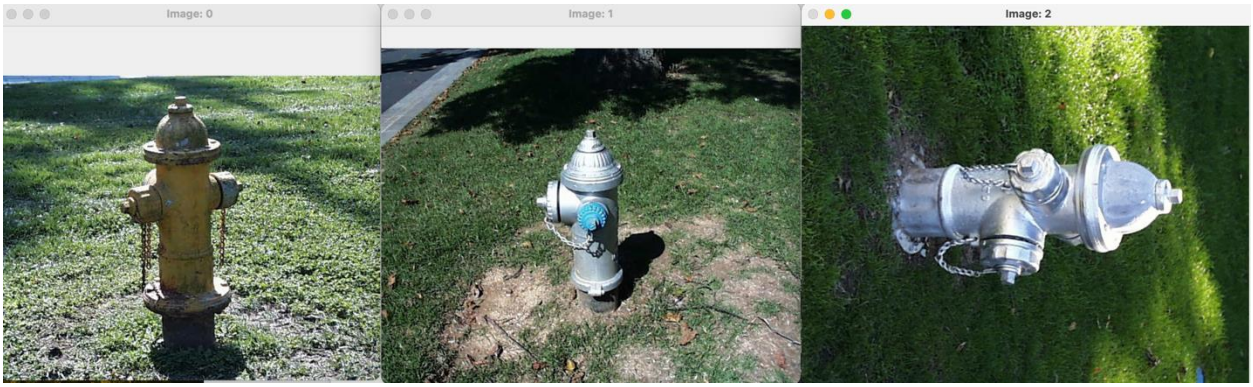
These are the best results using Multi RGB histogram



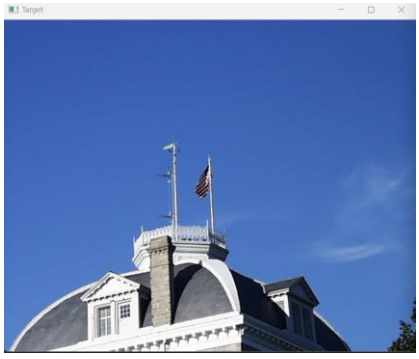
In these pictures it is evident that the addition of texture helps match the wall better as seen in the 2nd best match compared to the color histograms which seem more focused on the white in the chairs.

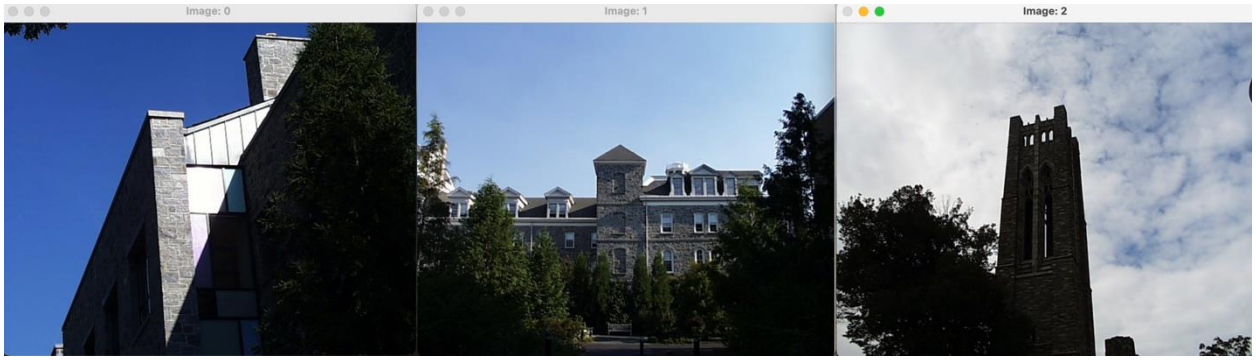
5. Deep Network Embeddings

For this task the feature vectors were provided in a csv. The feature vector contains 512 values derived from the final average pooling layer of a ResNet18 deep network pre-trained on ImageNet. ImageNet is a 1M image database with 1k categories of diverse types. The distance metric used was cosine distance because it was giving a better output because it takes into account the orientation as well. Below is the output for target image pic.0893.jpg



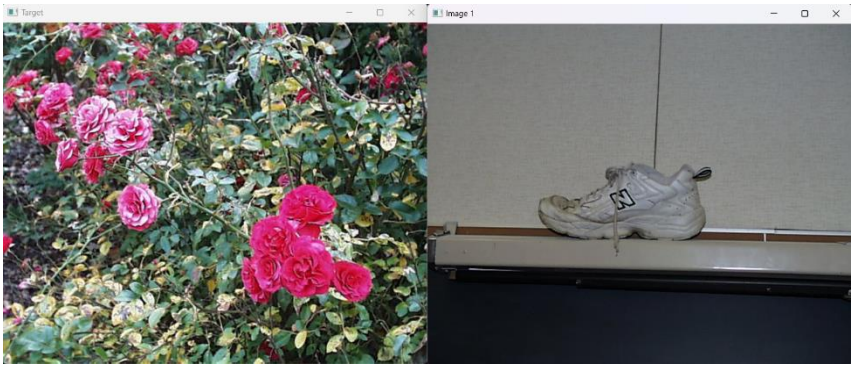
Output for target image pic.0164.jpg



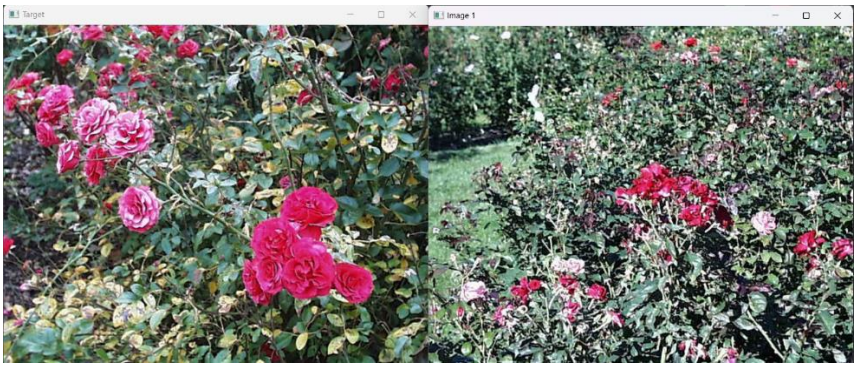


6. Which is better? DNN or Classic?

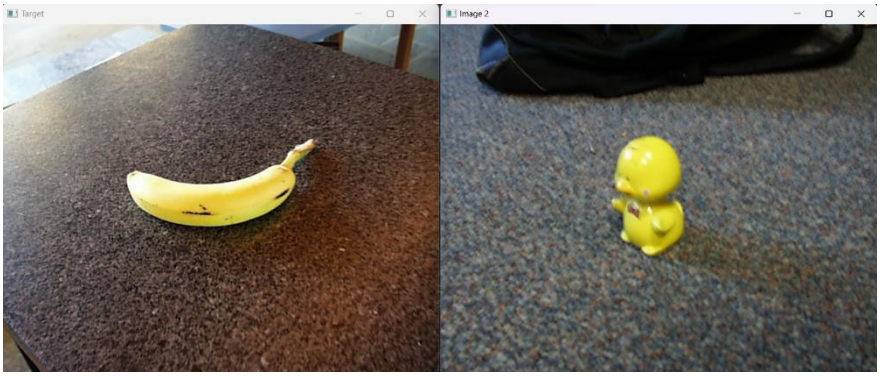
We compared the DNN embeddings in with 2 classical features. The first one is from task 1 using ssd. The second one is from task 7 using the custom weighted distance metrics.



Target image pic.1072.jpg
Using Baseline matching

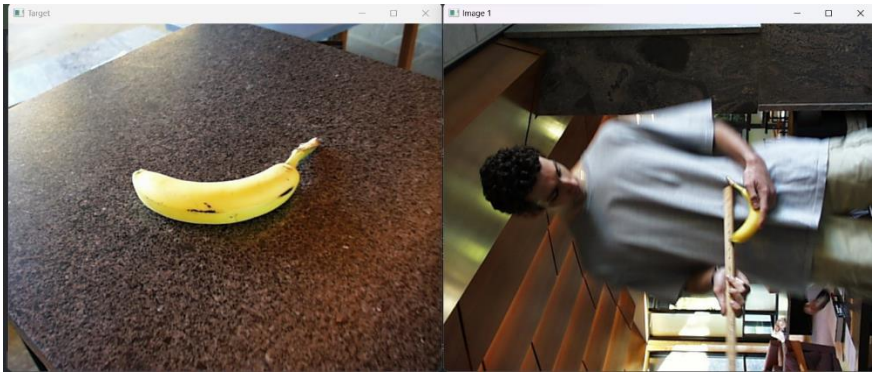


Target image pic.1072.jpg
Using DNN matching



Target image pic.0343.jpg

Using DNN matching



Target image pic.0343.jpg
Using Custom matching

From the above comparisons it is clear that DNN is not always better. It highly depends on the data it has been provided. However, DNNs can understand features such as objects which makes them very robust, provided they have been trained right. In cases where there is a repeating pattern in the color, texture and frequency space, it might be better to stick to generalized features which could save the costs and computation of training a model.

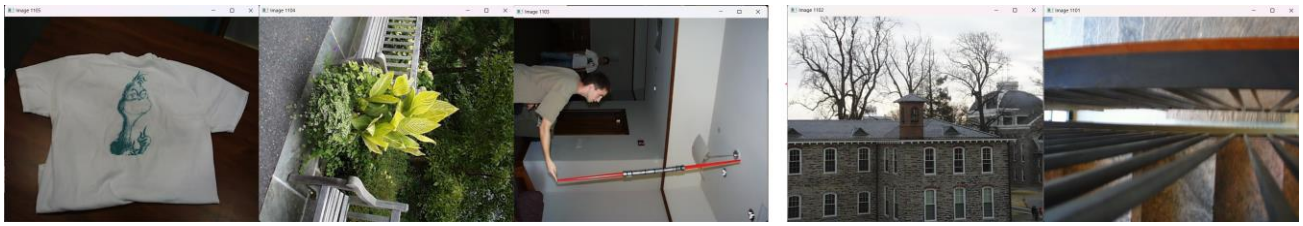
7. Custom Feature

The custom feature we decided on combines multiple image features—HSV histograms for color diversity, Gabor filter responses for texture analysis, and Fourier transform for frequency content—to create a robust image comparison framework. By calculating the cosine distances between Fourier and Gabor features, and the Chi-squared distance for HSV histograms, and integrating them through a weighted sum, this method offers a nuanced approach to image matching. This strategy leverages the complementary strengths of different feature sets, enhancing the ability to distinguish images based on color, texture, and frequency details. The approach is flexible, allowing adjustments in feature weighting to tailor the analysis to specific requirements, thereby improving the accuracy and reliability of image matching across diverse applications.

The matches for the target image pic.0343.jpg are as follows:

Top 5 Matches

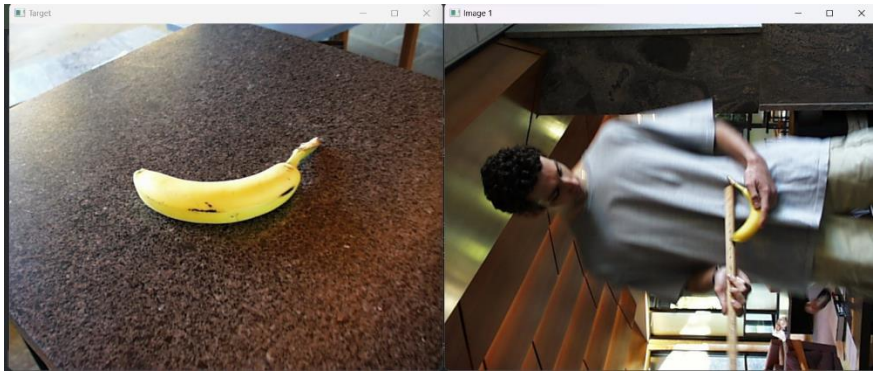
Bottom 5 Matches



The matches for the target image pic.0287.jpg are as follows:

Top 5 Matches

Bottom 5 Matches



From the first matching it is evident that the texture of the granite played a huge role in the resulting images. The best results are the ones that contain similar color schemes, similar textures and the same amount of details. The top result for the custom feature does beat the top DNN match.

For the second matching there isn't much texture, flatter colors and very less details all of which is also reflected in the resulting images.

Extensions

We proceeded with the following extensions

- Additional feature extraction methods
    - Function to compute Gabor Features
    - Function to compute Fourier Transform Features
    - Function to compute HSV histogram
- Additional Matching Methods
    - Bhattacharyya Distance
    - ChiSquared Distance

- Option show N least similar matches

Reflection

We learned a lot about the nuances of feature extraction.
For example, we understand that color histograms talk about the color space in an image. It does not look at the shape or texture of the object, it will give an image with the same amount of color in the target image for example it might match a sky and an ocean in a picture.
We understood that edge information helps understand the texture of the image and helps match repetitive patterns.
We also learned that Fourier transforms are useful in finding intricate details such as a decal or a mosaic in an image. The frequency information helps identify intricate details that might not be repeated as much.
We learned about matching methods and how they can be normalized to combine matching scores that are better higher or better lower in an overall weighted metric.

This project also delves heavily into the implementation of the process of extracting features from images - we chose a certain way to extract those features, particularly getting them in the form of vector<float> where each row is an image in the image data base. This choice informed a lot of the ways of implementing our different feature extraction methods. We mixed and matched a lot of feature types with distance metrics too to get a better sense of how the choice of distance metrics against our data affects the results. This project highlighted the trade-offs between different approaches, such as the simplicity and speed of classic features versus the complexity and depth of information captured by deep network embeddings.

References

https://stackoverflow.com/questions/2896600/how-to-replace-all-occurrences-of-a-character-in-string
https://safjan.com/metrics-to-compare-histograms/
https://mpatacchiola.github.io/blog/2016/11/12/the-simplest-classifier-histogram-intersection.html
Wikipedia
ChatGPT (To understand effect of Fourier transform on images also got the idea to shift quadrants from there)