CS 4186 Assignment 1
LIANG Qiyuan 54777209

Method 1: Color Histogram

1. Algorithm Description

For the color histogram method, I choose to implement the algorithm mentioned in the lecture, Swain and Ballard's Histogram Matching.

Step 1: Opponent encoding for each pixel: wb = R+G+B, rg = R-G, by = 2B-R-G
Step 2: wb is divided into 8 bins, each value of rg and by is divided into 16 bins. Total bins: 8*16*16=2048 bins
Step 3: Each pixel is mapped to the 2048 bins, and a color histogram is generated.
Step 4: Calculate the intersection of the query image and the images in the database using the formula: **Intersection(h(I),h(M)) =** $\sum_{j=1}^{numbins} \mathbf{min}\ \{\boldsymbol{h}(\boldsymbol{I})[\boldsymbol{j}], \boldsymbol{h}(\boldsymbol{M})[\boldsymbol{j}]\}$
Step 5: Normalize the match score, where M is the query image:
match(h(I),h(M)) = Intersection(h(I),h(M))/ $\sum_{j=1}^{numbins} \boldsymbol{h}(\boldsymbol{M})[\boldsymbol{j}]$
Step 6: Sort the images based on match score.

2. Result Analysis

Mean Average Precision: **0.073896**

Details of the 10 example queries:
Average Precision of Q1: 0.0267
Average Precision of Q2: 0.0042
Average Precision of Q3: **0.3658**
Average Precision of Q4: 0.0110
Average Precision of Q5: 0.0042
Average Precision of Q6: **0.2611**
Average Precision of Q7: 0.0038
Average Precision of Q8: 0.0115
Average Precision of Q9: 0.0420
Average Precision of Q10: 0.0087

High Precision Queries

The precision of Q3 and Q6 are reasonable, let's look at them in detail.

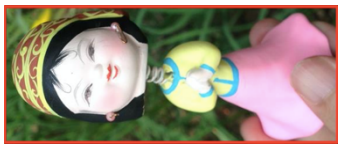
Q3: Query image

Top-6 ranking images:

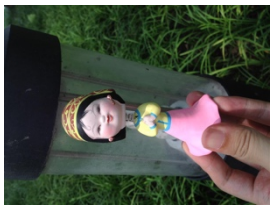





Q6: Query image

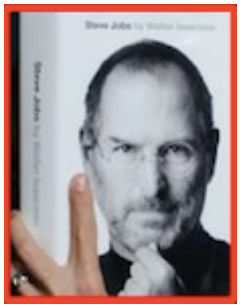


Top-6 ranking images:

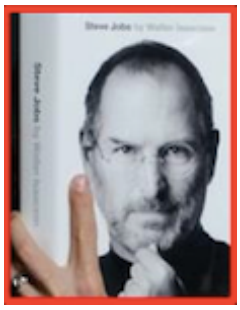

From the results of Q3 and Q6, I could find out that the top-ranking images have similar colors to those in the query image.

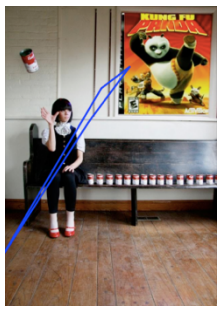
Poor Precision Queries



I could find out that, for the poor performance queries, they either have distracting backgrounds or color is not a good indicator for the query instance.

Examples that color is not a good indicator and the background is distracting







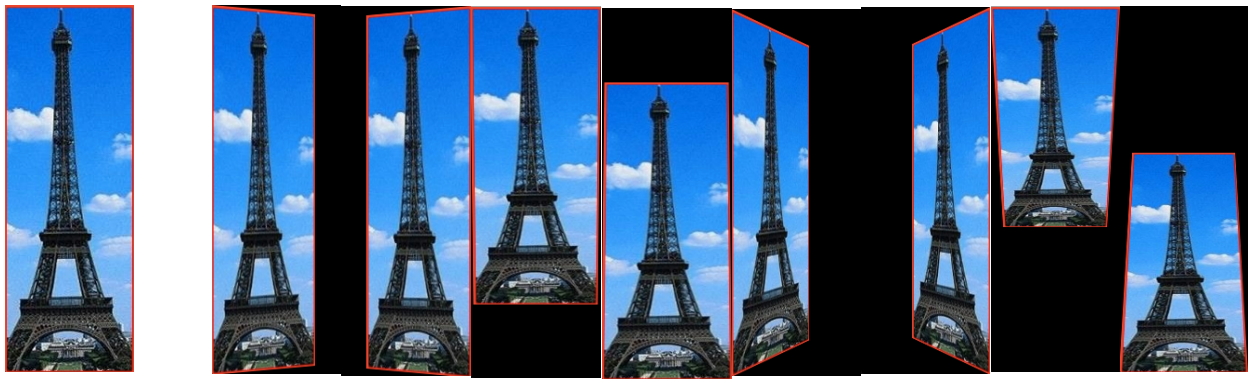

Method 2: SIFT + Homography

1. Algorithm Description

For the second method, I choose SIFT (Scale-Invariant Feature Transform) to extract features, and apply feature matching and homography to find objects.

Step 1: Process the images in the database, detect corners and use SIFT to describe features inside each image.
Step 2: Perform a data augmentation on the query images. Apply eight different perspective transformation of each query image to enhance the chance of matching in the database.

Query image tansformed images



Step 3:
For each of the nine images generated in step2, it will go through the following steps:

1) Detect corners and use SIFT to describe features;
2) Use brute force matcher to match the features in 1) with the features of each image inside the database;
3) Find the perspective transformation of the object and filter out incorrect matches;

Step 4: The maximal number of correct matches of the nine images found in Step 3 is assigned as the score of the similarity between each query image and each image inside the database.
Step 5: Sort the images based on the match score.

2. Result Analysis

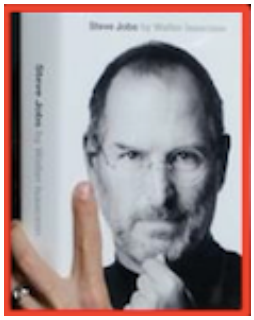
Mean Average Precision: **0.427257**

Details of the 10 example queries:
Average Precision of Q1: 0.2509
Average Precision of Q2: **0.8836**
Average Precision of Q3: **1.0000**
Average Precision of Q4: 0.0302
Average Precision of Q5: 0.3958
Average Precision of Q6: **0.9488**
Average Precision of Q7: 0.0050
Average Precision of Q8: 0.4727
Average Precision of Q9: 0.0254
Average Precision of Q10: 0.2602

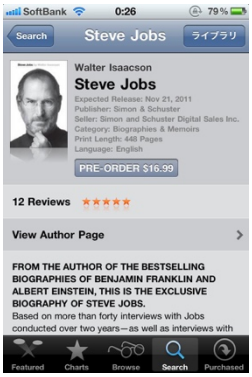
High Precision Queries

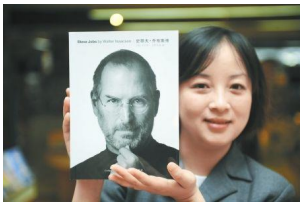
The precision of Q2, Q3, and Q6 are good, let's look at Q2 in detail.

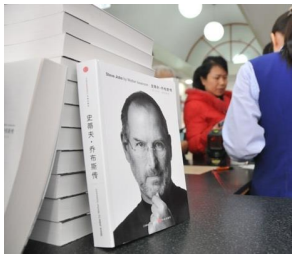
Q2: Query image

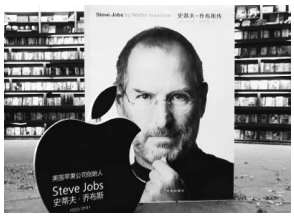


Top-6 ranking images:





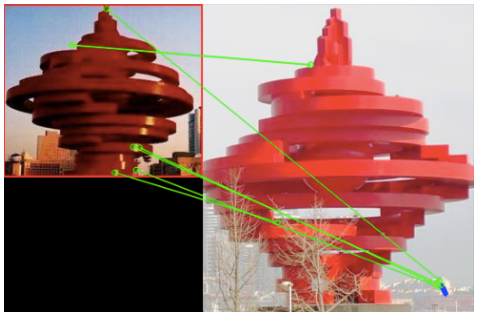




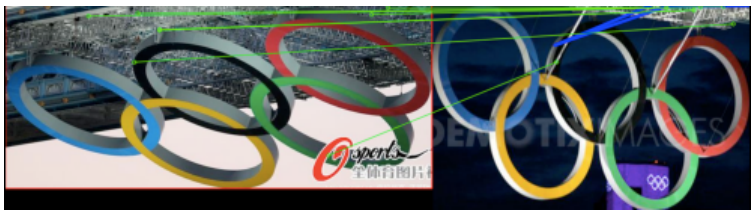
Poor Precision Queries

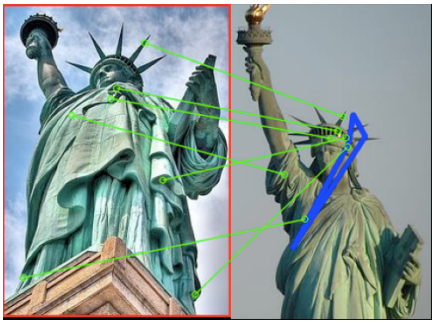
For the poor precision queries, they either have a very different perspective compared to those in the database or distracting background that produces unwanted corners.

Examples of distracting background



Examples of very different perspectives





Conclusion

For method 1, it is not as robust as method 2. When the color of the image is a good indicator of the object, it works pretty well. However, when the color is not very unique or the image contains distracting background, color histogram gives poor performance.

For method 2, it is much more robust. It is scale and rotation invariant, and can handle small changes in viewpoint. However, when the change in perspective is too large or the background of the query image is not clean, the performance becomes poor.