REPORT ON
FEATURES AND IMAGE MATCHING

Name: Trần Trường Giang
Student ID: BI9-090

I. Objective

Matching features of the same object in different images. Features can be used for many applications:

- Image alignment
- Objective recognition
- Motion tracking
- Navigation

II. Method

The process of matching image are separated to 2 major step:

1. Finding feature

First we need to find the interest point of the image. Imagine each interest point is a window on the image.A good interest point needs to be unique, which means we can easily detect change when the window is moving. For this reason we should choose a corner in an object rather than an edge or a flat region. To do this we can use Harris operator to map out the Harris value for each pixel, then use a threshold to limit which point feature point that you want to use based on the fact that the higher the value is the more it is likely to be a corner.

Opencv has many features descriptor built in but in this report I am going to dive deep into SIFT, because it gives me the best result and ORB because it is not patterned, hence, free to use.

2. Matching feature

In this step we can use brute-force matcher or use a FLANN based matcher.

Brute-Force matcher takes the descriptor of one feature in the first set and is matched with all other features in the second set using some distance calculation. And the closest one is returned. Second param of the function is crossCheck, used to provide a consistent result. If it is true, Matcher returns only those matches with value (i,j) such that i-th descriptor in set A has j-th descriptor in set B as the best match and vice-versa.

FLANN stands for Fast Library for Approximate Nearest Neighbors. It contains a collection of algorithms optimized for fast nearest neighbor search in large datasets and for high dimensional features. It works faster than BFMatcher for large datasets.

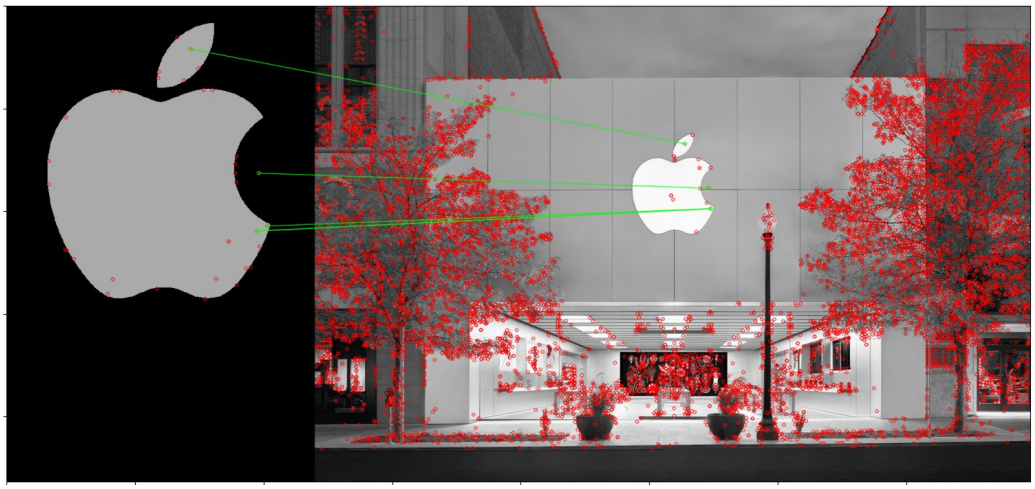
III. OpenCV Illustration

For this example I am going to use SIFT detection with FLANN based matcher to match this Apple logo on to the store picture.

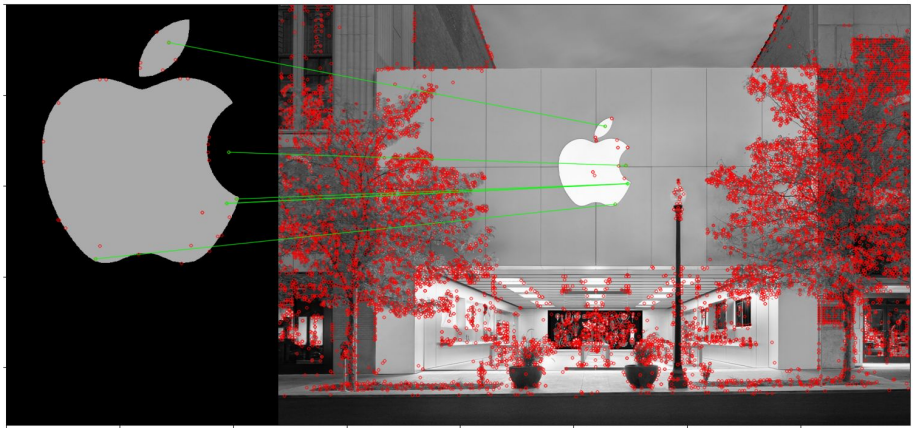
Code:

```
import numpy as np
import cv2 as cv
import matplotlib.pyplot as plt
img1 = cv.imread('apple-logo.png',cv.IMREAD_GRAYSCALE)  # queryImage
img2 = cv.imread('apple-store.jpg',cv.IMREAD_GRAYSCALE) # trainImage
# Initiate SIFT detector
sift = cv.SIFT_create()
# find the keypoints and descriptors with SIFT
kp1, des1 = sift.detectAndCompute(img1,None)
kp2, des2 = sift.detectAndCompute(img2,None)
# FLANN parameters
FLANN_INDEX_KDTREE = 1
index_params = dict(algorithm = FLANN_INDEX_KDTREE, trees = 5)
search_params = dict(checks=50)   # or pass empty dictionary
flann = cv.FlannBasedMatcher(index_params,search_params)
matches = flann.knnMatch(des1,des2,k=2)
# Need to draw only good matches, so create a mask
matchesMask = [[0,0] for i in range(len(matches))]
# ratio test as per Lowe's paper
for i,(m,n) in enumerate(matches):
      if m.distance < 0.7*n.distance:
      matchesMask[i]=[1,0]
draw_params = dict(matchColor = (0,255,0),
            singlePointColor = (255,0,0),
             matchesMask = matchesMask,
            flags = cv.DrawMatchesFlags_DEFAULT)
img3 = cv.drawMatchesKnn(img1,kp1,img2,kp2,matches,None,**draw_params)
plt.imshow(img3,),plt.show()
```

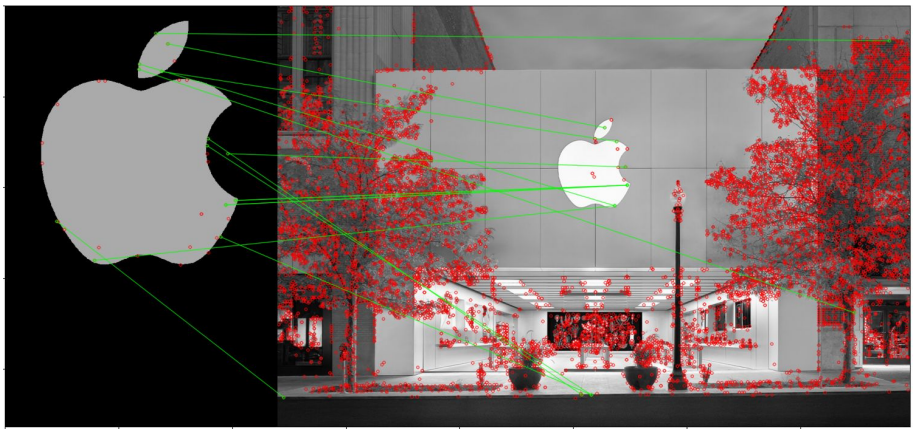
Result:



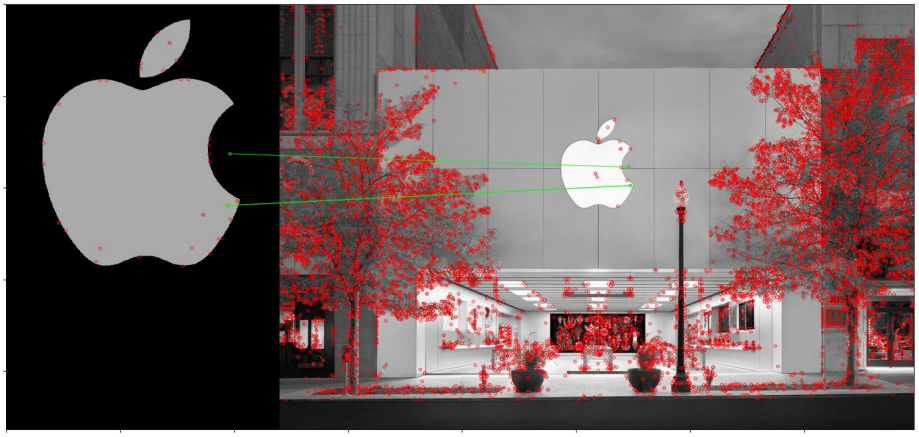
When we change the threshold ratio to 0.8 we can see that the algorithm starts to have some incorrect match.



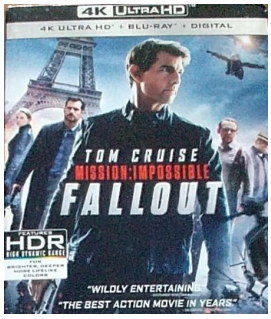
When we change the threshold ratio to 0.9 the algorithm shows even more incorrect mistakes just as expected.

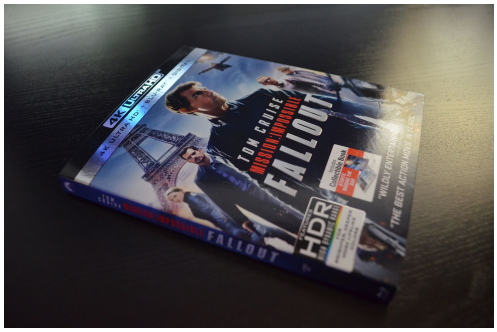


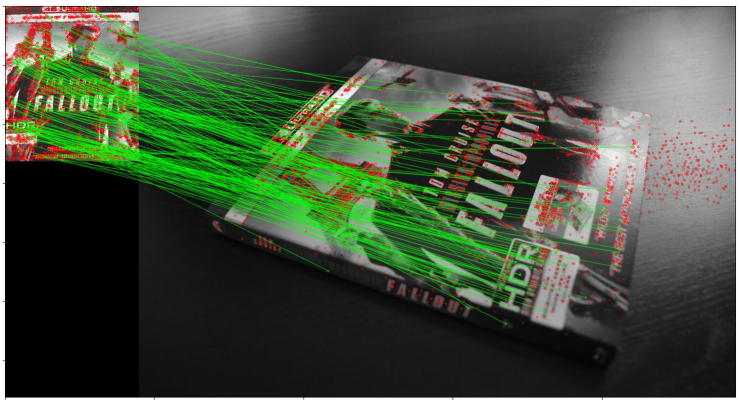
And if we change the threshold ratio to 0.65 the reverse is happening, the algorithm shows less matches but it only shows accurate ones.



Here is an example of an image with more detail:







In the example above, in ratio test step, the first threshold was set to 0.7 and the second threshold was set to 0.6. Even though almost half of the correct matches got eliminated we can still detect some obvious mistakes by checking manually.

IV. Conclude

To show different results of different feature matches in accuracy, we can manipulate the ratio threshold in the ratio checking step. The higher the ratio setted, the lower the accuracy becomes.