Lab 08

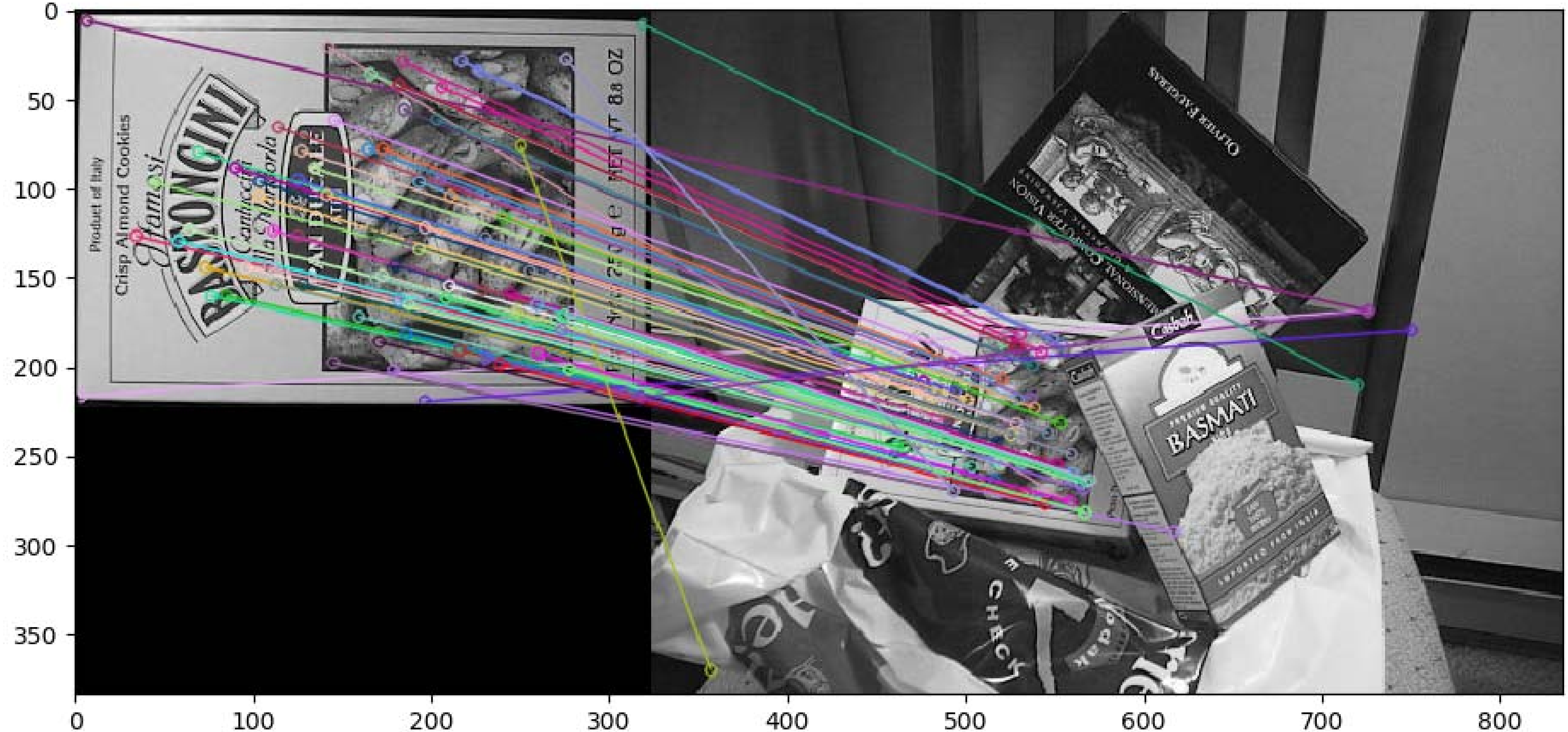
SIFT feature matching

opencv

- Finally we are using opencv
- Pip install opencv-contrib-python
  - Opencv-python does NOT contain SIFT

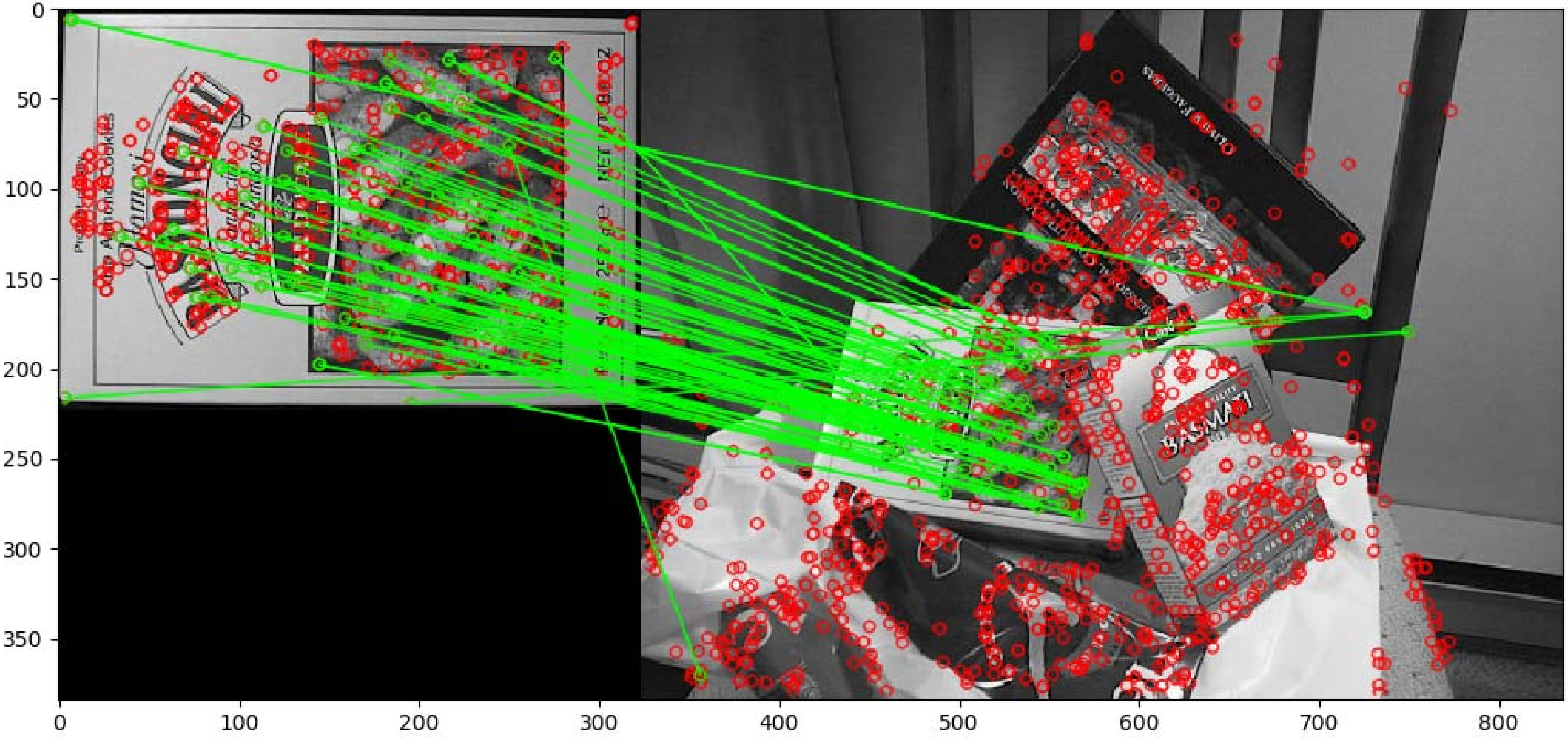
SIFT

```
import numpy as np
import cv2
from matplotlib import pyplot as plt

img1 = cv2.imread('box.png',0)          # queryImage
img2 = cv2.imread('box_in_scene.png',0) # trainImage

# Initiate SIFT detector
sift = cv2.SIFT()

# find the keypoints and descriptors with SIFT
kp1, des1 = sift.detectAndCompute(img1,None)
kp2, des2 = sift.detectAndCompute(img2,None)

# BFMatcher with default params
bf = cv2.BFMatcher()
matches = bf.knnMatch(des1,des2, k=2)

# Apply ratio test
good = []
for m,n in matches:
    if m.distance < 0.75*n.distance:
        good.append([m])

# cv2.drawMatchesKnn expects list of lists as matches.
img3 = cv2.drawMatchesKnn(img1,kp1,img2,kp2,good,flags=2)

plt.imshow(img3),plt.show()
```


0
50
100
150
200
250
300
350
0
100
200
300
400
500
600
700
800
BASTONCINI
Famosi
Crisp Almond Cookies
Product of Italy
PAN DI CALE
NET WT 8.8 OZ
OLIVER FAUGERAS
BASMATI
Casbah

FLANN-based matching

```
# FLANN parameters
FLANN_INDEX_KDTREE = 0
index_params = dict(algorithm = FLANN_INDEX_KDTREE, trees = 5)
search_params = dict(checks=50)   # or pass empty dictionary

flann = cv2.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 = 0)

img3 = cv2.drawMatchesKnn(img1,kp1,img2,kp2,matches,None,**draw_params)

plt.imshow(img3,),plt.show()
```


0
50
100
150
200
250
300
350
0
100
200
300
400
500
600
700
800

homework

- Write a python code that
  - Read an input image (big)
  - Find SIFT keypoints
  - Match with bruteforce / FLANN and compare
- Write a report with procedure and comparison
- Submit your
  - Code
  - Report
  - Input image
  - Matching result