CS5542 Big Data Analytics and App

Lab Assignment #1

Submitted by:

Koushik Katakam – 10

Team – 5

Goals:

The goals of the Lab Assignment 1:

1. Based on the Project theme, download the dataset.
2. Perform basic NLP operations namely Tokenization and Lemmatization on the downloaded captions.
3. From the extracted data, report the image statistics.
4. Perform feature extraction using SIFT algorithm on the image data.

Technologies Used:

Pycharm

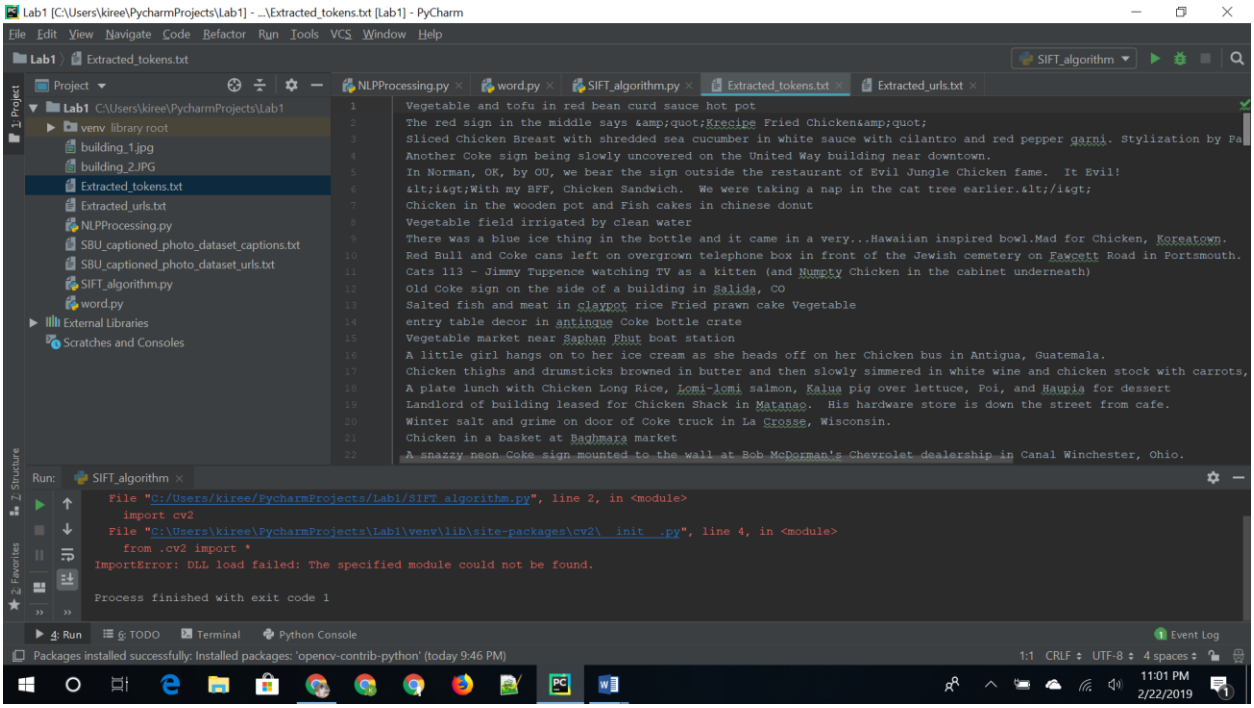
Packages Used:

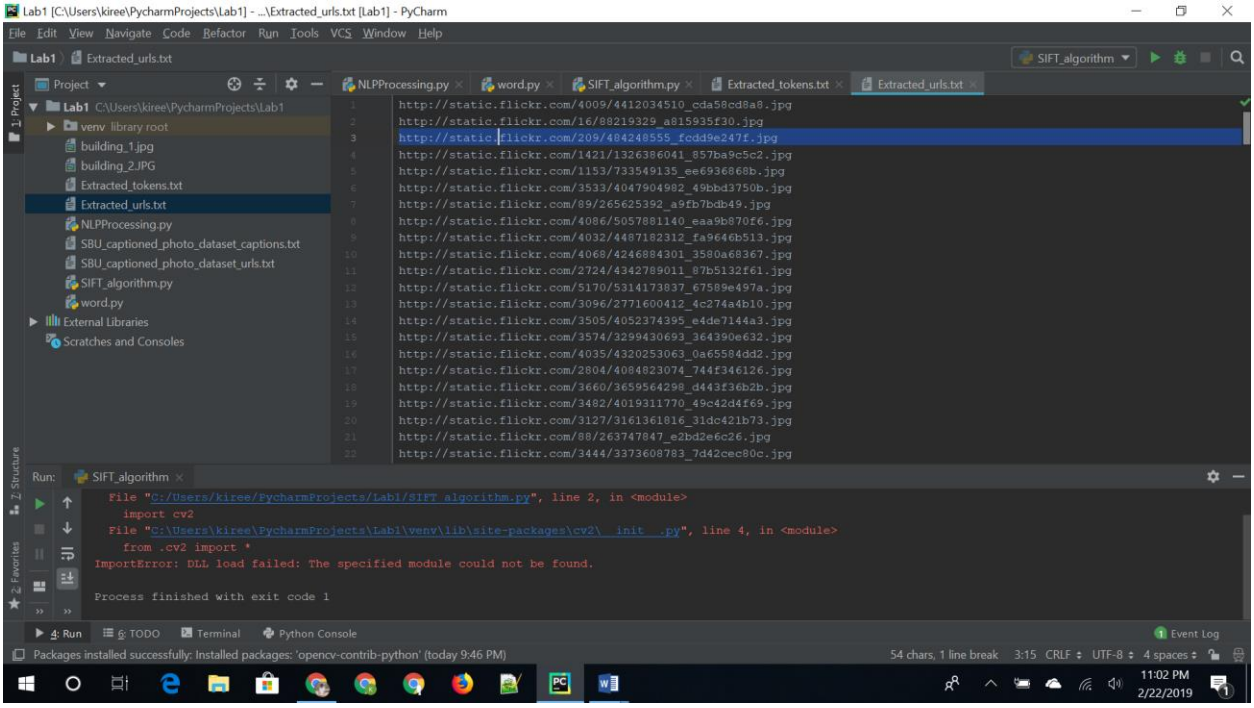
- nltk
- opencv-python
- numpy
- matplotlib
- tensorflow
- linecache

Dataset:

There are many datasets available online. From those, we have chosen SBU dataset in which the data is in the form of “URL’s” and “Captions”.

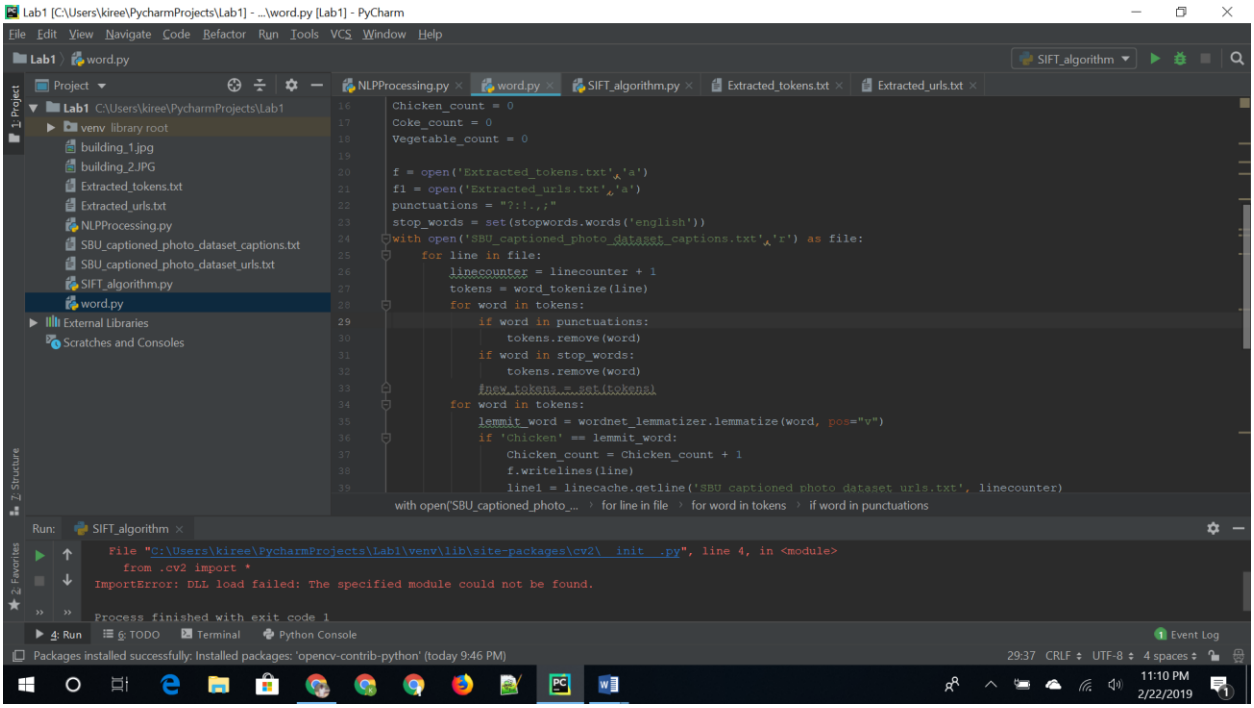
Screenshots of Dataset URL's and corresponding Captions:





Tokenization:

- Tokenization generally tokenizes the sentence or a paragraph into corresponding words and stores them in the form of list.
- There are two ways to do Tokenization namely word tokenization(word_tokenize) and sentence tokenization(sent_tokenize).
- Word Tokenization is used for extracting the required captions from the dataset.
- Using linecache package the urls are extracted after the process of tokenization and store them in the new file.



Lemmatization:

- Lemmatization generally produces the output of the word in its root form which can be either in the form of adjective, noun, verb.
- We eliminate stopwords when we do lemmatization process.

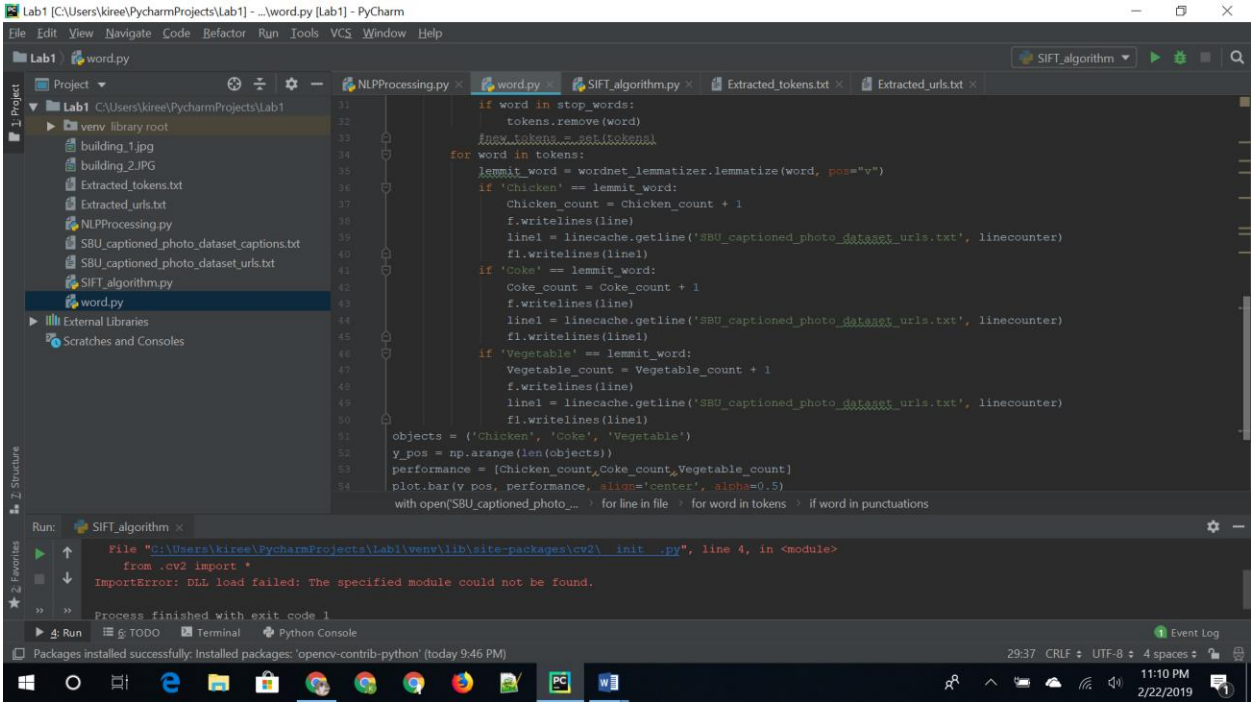
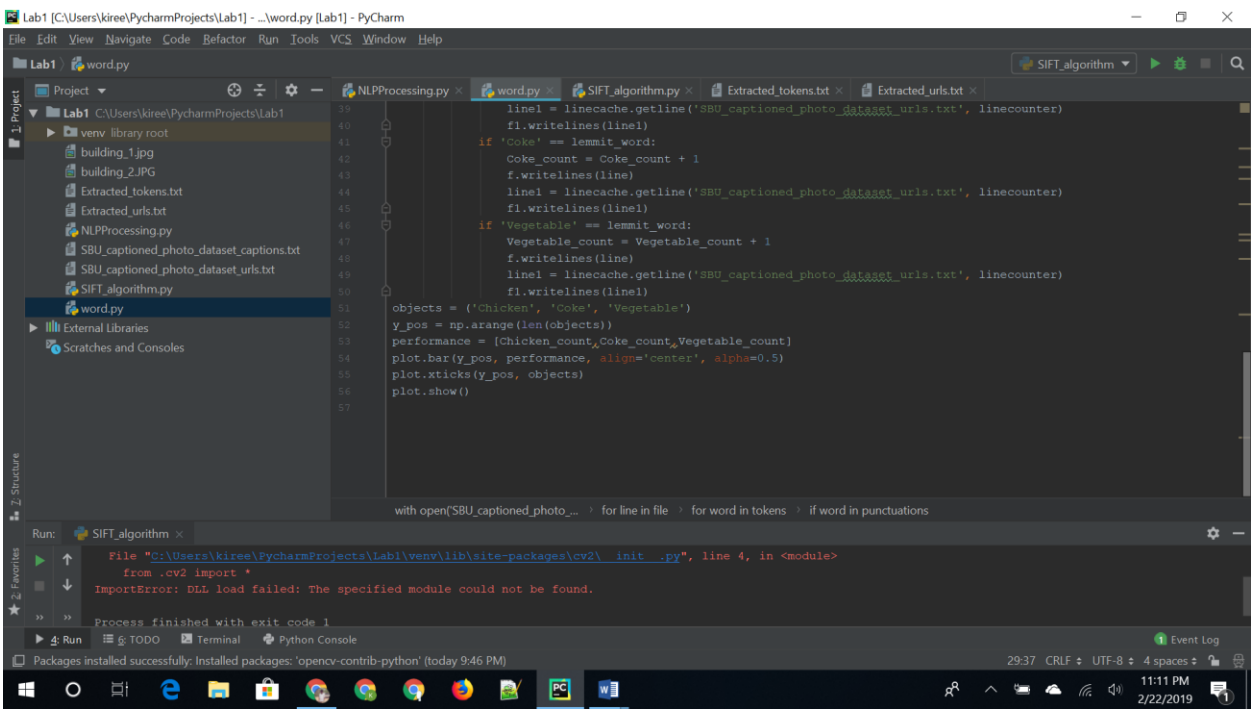
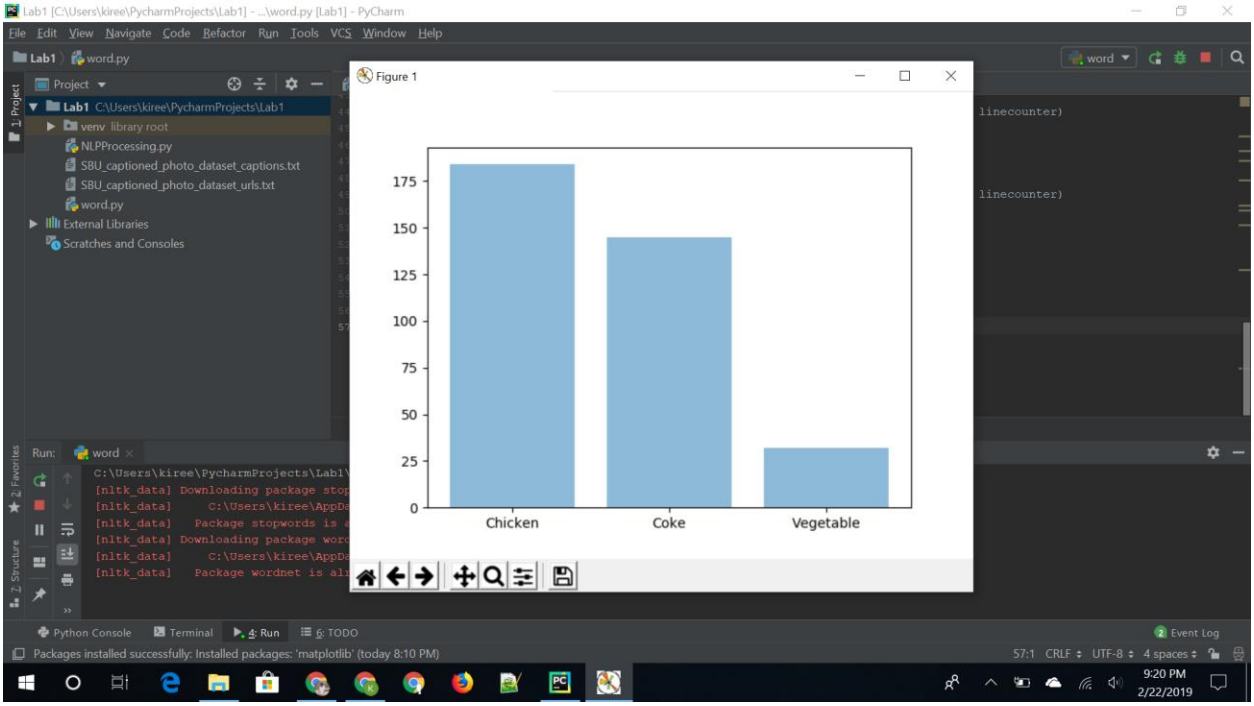
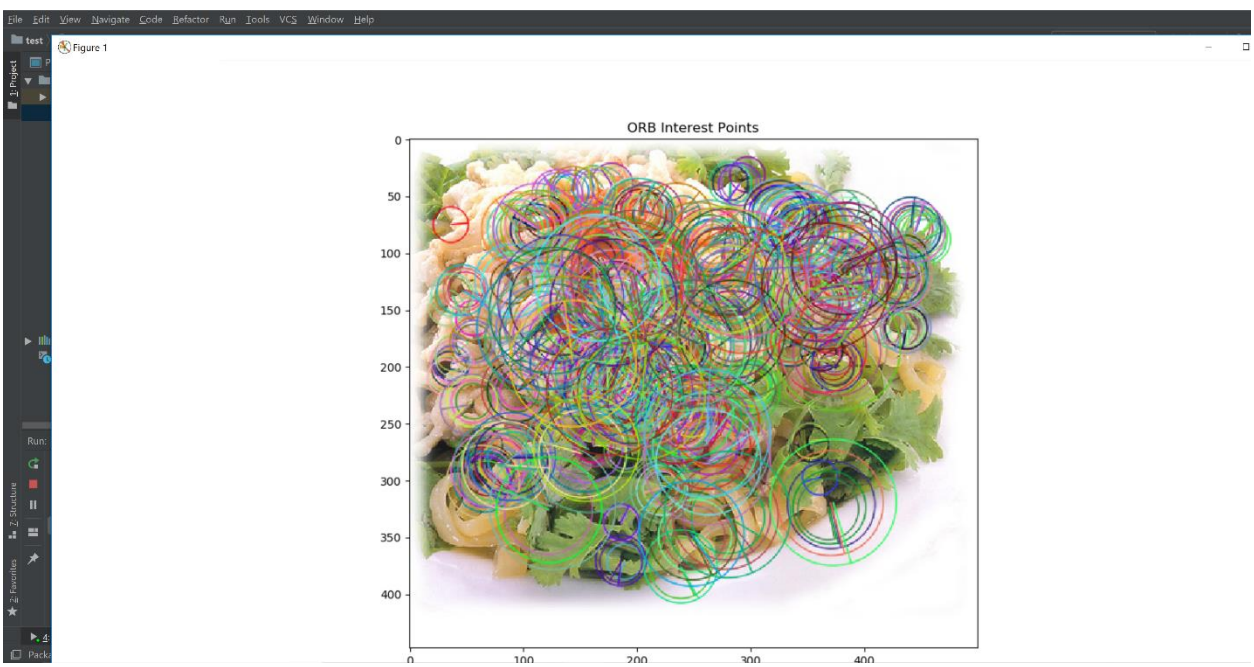


Image Statistics:





SIFT Algorithm:



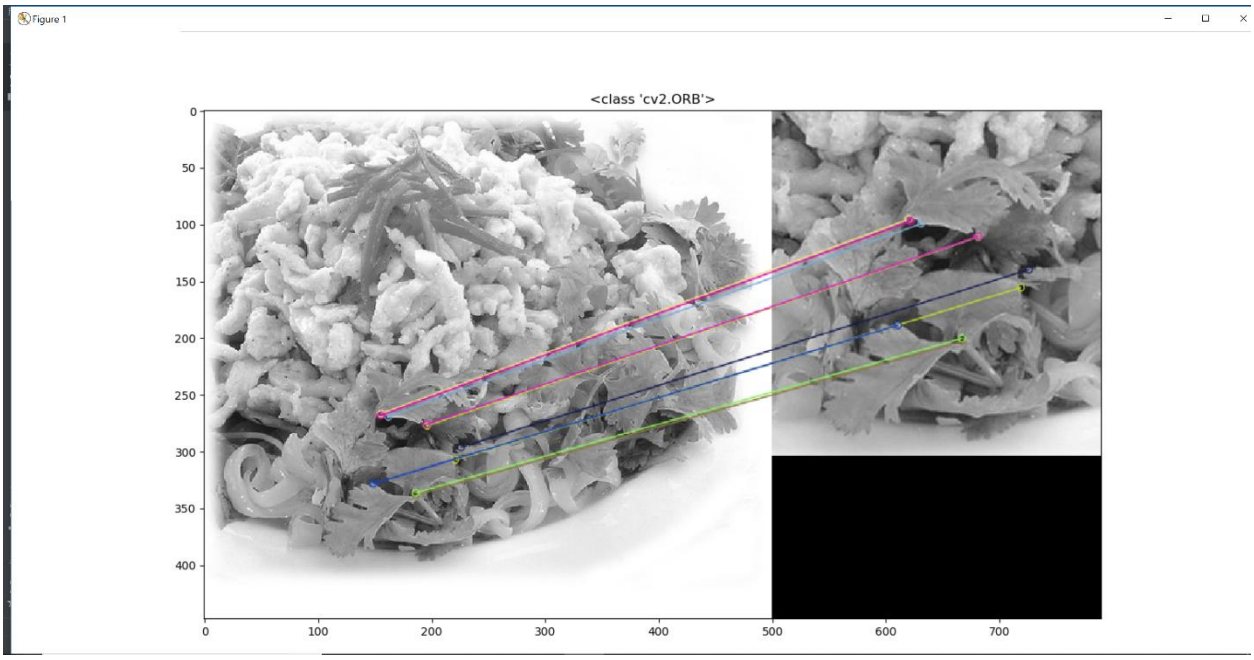


Figure 1
<class 'cv2.ORB'>

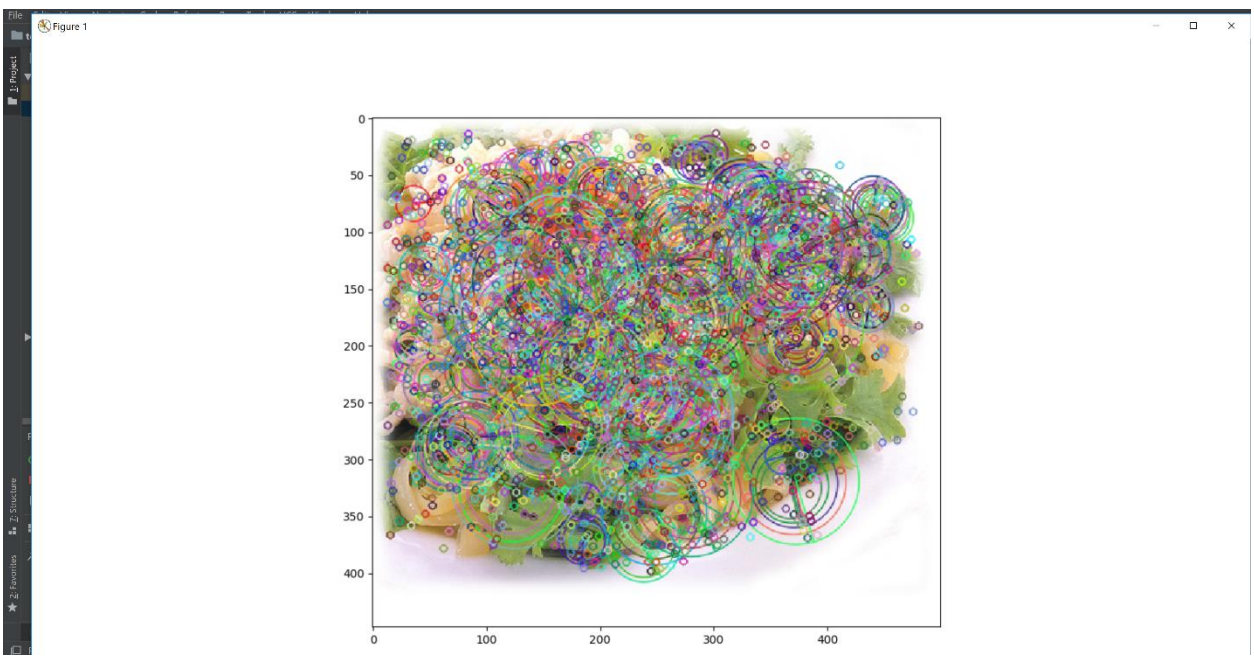


Figure 1

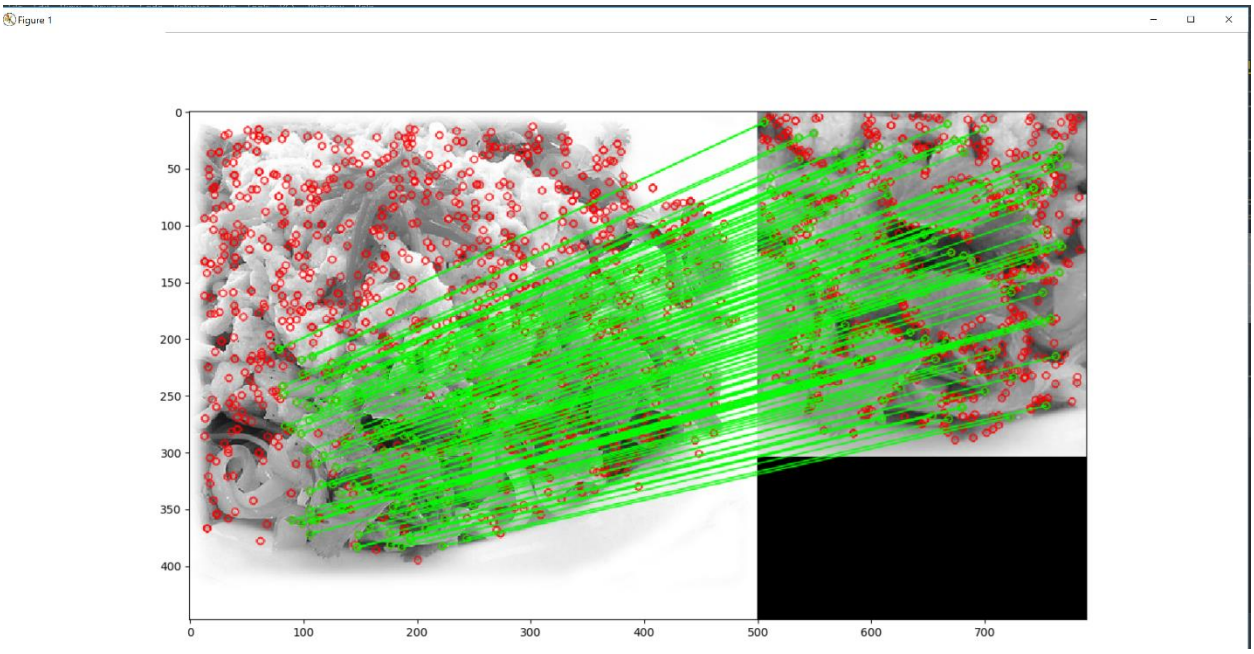


Figure 1
0
50
100
150
200
250
300
350
400
0
100
200
300
400
500
600
700


Refactor Run Tools VCS Window Help
SIFT_algorithm
SIFT_algorithm.py
44 plt.title(type(detector))
45 plt.imshow(img_matches)
46 plt.show()
47
48
49 orb = cv2.ORB_create()
50 draw_image_matches(orb, 'building 1.jpg', 'building 2.jpg')
51
52 sift = cv2.xfeatures2d.SIFT_create()
53 kp, des = sift.detectAndCompute(img_building, None)
54 img_kp = cv2.drawKeypoints(img_building, kp, img_building)
55
56 plt.figure(figsize=(15, 15))
57 plt.imshow(img_kp); plt.show()
58
59 img1, kp1, des1 = image_detect_and_compute(sift, 'building_1.jpg')
60 img2, kp2, des2 = image_detect_and_compute(sift, 'building_2.jpg')
61
62 FLANN_INDEX_KDTREE = 1
63 index_params = dict(algorithm = FLANN_INDEX_KDTREE, trees = 5)
64 search_params = dict(checks=50)
65
66 flann = cv2.FlannBasedMatcher(index_params,search_params)
67 matches = flann.knnMatch(des1, des2, k=2)