Assignment 2 Report

Harsh Vora
CWID : A20445400
CS512 – F20

October 20, 2020

1. Problem Statement

- Capture or load two images that contains similar content and process the images continuously.
- The input image should always be converted to grayscale before processing it.
- Make the program interactive by including manipulations either through keyboard/mouse/trackbar.
- Program must include a help key describing its functionality.
- To estimate image gradients and apply the Harris corner detection Algorithm.
- To compute a feature vector for each corner point.
- Using the feature vectors computed match the feature points. Number corresponding points with identical numbers in two images.
- Interactively controlled parameters should include:
  1.The variance of the Gaussian(scale)
  2. The neighborhood size for computing the correlation matrix
  3. The weight of the trace in Harris corner Detector.
  4. Threshold Value.

2. Proposed Solution

- First, the program takes two images either by capturing or it checks the arguments. If the arguments do not contain the name of the image it will automatically capture the image through camera.
- After taking the images it will convert the images to grayscale.
- After converting the images into grayscale, it will apply sobel filter both in x and y axis and then Ixx, Iyy, Ixy is computed for each pixel in the image.
- After that a correlation matrix is calculated in each window of the image with the values of Ixx, Iyy and Ixy.
- After constructing the correlation matrix, the determinant and the trace is calculated.
- Determinant and trace are used to calculate C(G)= det – k *(trace **2) value of each pixel which is then stored in a list along with the pixel position.
- Then we check if the value of C(G) is greater then the threshold provided by the user then the pixels are selected as a corner and then rectangle are drawn around those pixels.
- After this to compute the feature vectors I made use of ORB descriptors and then found the keypoints and descriptors of both image with SIFT.

- After that I used BFmatcher to match the descriptors of both the images and then sorted them in the order of distance.
- After that from all the points queryIdx and trainIdx is stored in 2 different lists.
- Then from that list the number corresponding points with identical numbers in the image is shown.
- After that for better localization I projected gradients onto edge hypothesis and then I chose p with minimal projection.

3. Implementation Details

- The program will capture the image from the camera and will give 2 similar images.
- Implemented a display function to display the results.
- I did not use trackbar to implement the real time parameter as it was taking too much time to complete the rebuild of functions, Therefore I chose Keyboard to interact with the program as it will take the parameter from the user from first and then show the corners on the image**.**
- Implemented a key function where pressing “h” will show what keys needs to be pressed to perform different functionalities.
- The user should provide all the parameters to get results for Corner Detection.
- All the parameters must be selected depending on the texture and the complexity of the image.

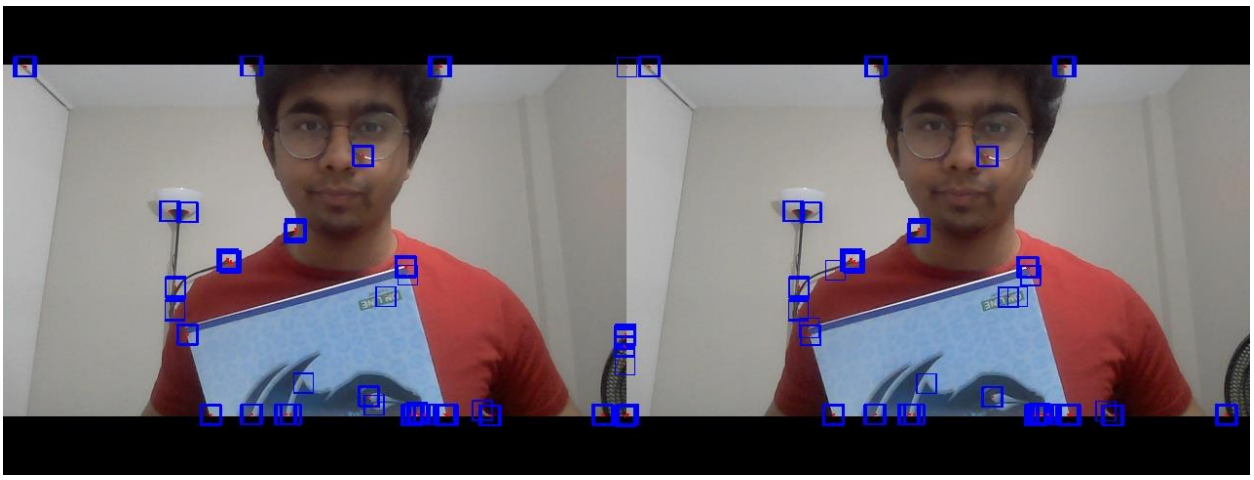
4. Results and Discussion

- Image can be loaded either via camera or can be given during the argument while running through the command line.

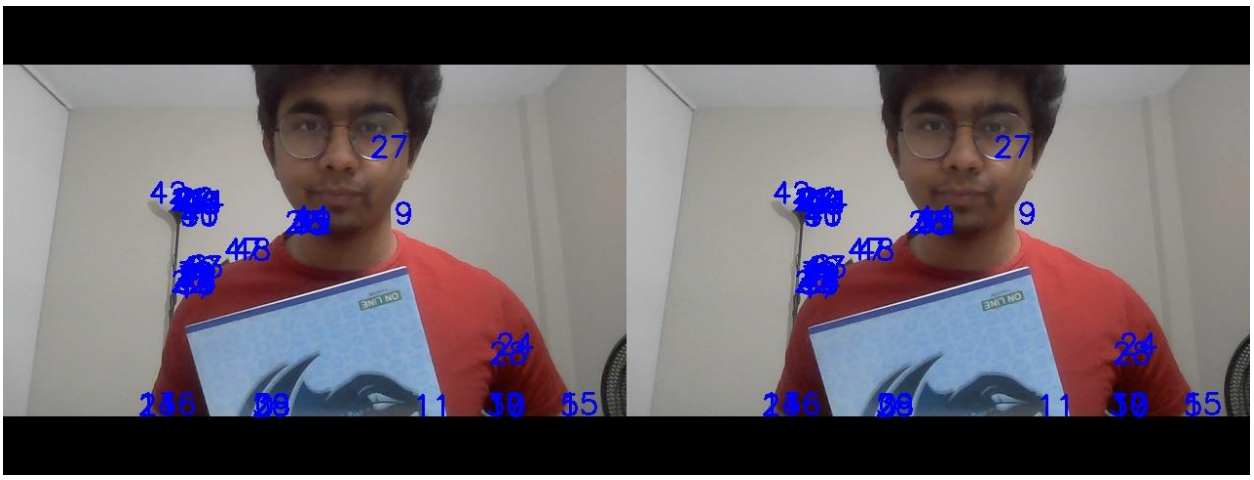
- This are the parameters that user must input for performing Corner Detection:

```
input key to Process image(press 'H' for help, press 'q' to quit):
c
Enter the varience of Gaussian scale(n):3
Size of the Window :2
Enter the weight of the trace in the harris corner detector between (0, 0.5)(k):0.05
Enter Threshold Value:700000
Processing the images...
```

- Output of Corner Detection



- Output of Feature Vectors:



- Output of Better Localization:

- Output of the Help function in the program:

```
input key to Process image(press 'H' for help, press 'q' to quit):
H
'c': Estimate image gradients and apply Harris corner detection algorithm.
'b': Obtain a better localization of each corner.
'f': Compute feature vectors for each corner that were detected.
```