Student: **Ulvi Bajarani**

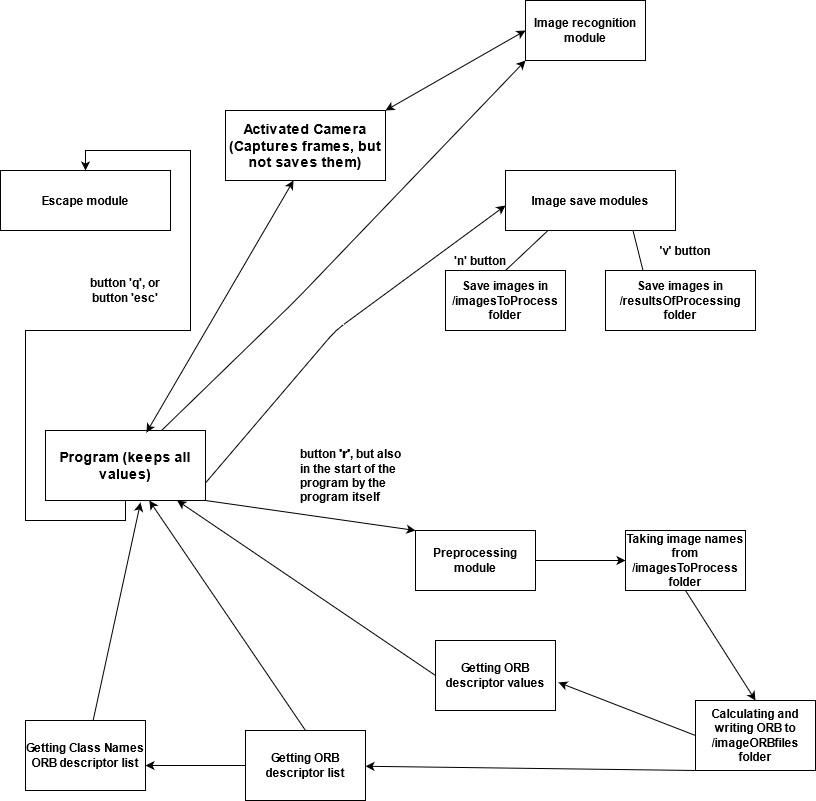
SID: 20539914

Title: Mini Project N4 – ORB (Oriented FAST and Rotated BRIEF)

March-4-2021

**Introduction**

Here, the schema of the project:



When the program executes, it already executes preprocessing module. It includes:

1) file\_list\_to\_process(folder\_path\_to\_process) – the function returns the list of available images from ***/imagesToProcess*** folder. Returns the list of images. If there is no image, returns an empty list.

2) calculateAndWriteDescriptors(image\_files\_list, folder\_path\_to\_process, folder\_path\_to\_write\_orb) – Removes all available ORB file descriptors and writes the ORB file descriptors from the available images in the **image\_files\_list**. If the length of the list is 0, does nothing. The file format is ORB that is simply CSV, but with ORB extension. Returns nothing.

3) orb\_list\_to\_process(folder\_path\_to\_write\_orb) – Gets the list of ORB to process. Returns the list of ORB. If the length of the list is 0, returns an empty list.

4) get\_class\_names\_from\_descriptors(image\_descriptions\_list) – Gets the class names from the names of ORB descriptors. Returns the list of class names. If the length of the list is 0, returns an empty list.

5) get\_class\_values\_from\_descriptors(image\_descriptions\_list, folder\_path\_to\_write\_orb) – Gets the class values from the names of ORB descriptors. Returns the list of class names. If the length of the list is 0, returns an empty list.

The important thing is that by button ‘r’, it is possible to execute preprocessing process again. It allows, in the case of renaming files, get new class names.

While the result is found, the program puts the text on the image with the proper class name. It is done by comparing the value of proper class (if it is not equal to -1) and by **cv2.putText(imgOriginal, class\_descriptors[ProperClass], (50,50), cv2.FONT\_HERSHEY\_PLAIN, 2, (0,0,255), 2)** function.

Another modules are capturing image modules. While by the button ‘n’ it is possible to get new images in the **/imagesToProcess** folder for the training, by pressing ‘v’ it is possible to save results in the **/resultsOfProcessing** folder. These modules have own counters and save the file as frame***OwnCounter***.png

The ORB is calculated by the Brute-Force Matching. More details are in the discussion.

To quit from the program, either ‘esc’ button or ‘q’ button must be pressed.

**Results**

|  |  |  |
| --- | --- | --- |
| Image Object | Object name | Query Result |
|  | Key (ORB saved as key.orb) |  |
|  | Phone (ORB saved as phone.orb) |  |
|  | Remote Controller (ORB saved as remote controller.orb) |  |
|  | Watch (ORB saved as watch.orb) |  |

**Discussion**

There were several issues:

1. In the KNN matching, the threshold could be chosen by the number of the matching. However, in the distance, descriptor has only distance. In this case, to be able to distinguish, the author decided to use the mean values of all distance to choose best matches, since the number of matches is not the best feature due to various possible sizes (for example, there might be 100 matches, but their distance are higher). The matching threshold is put as **mean < 50**.
2. While watch and remote controllers were identified easily, and the phone after some effort, the key identification was the hardest one.
3. The descriptors are calculated by bf = cv2.BFMatcher(cv2.NORM\_HAMMING, crossCheck=True), while matching is done by bf.match(des2, descriptor).
4. The image to compare is taken as grayscale.

**Conclusion**

The project might be enhanced by using proper GUI’s. Also, the resizing of images could be used to get best matches.