COMPUTER VISION - LAB 4

Computer Vision 2022,
P. Zanuttigh, M. Mel, A. Simonetto, M. Toldo

Topics: Keypoints, Descriptors and Matching
Goal: Create a mosaic image by joining together a group of images.

Write a C++ software which includes methods that:

1. Load a set of images from one of the provided datasets.
    a. The images are roughly arranged in a 3x3 grid and have been obtained by splitting a larger image and applying some transformations.
    b. Different sets have different level of complexity: the folder names underline the applied transformations: “T” stays for translation, “R” rotation, “S” scaling, “N” noise, “L” illumination change
    c. It is not required that you test or solve all the datasets, as expected the ones with more transformations inside are more challenging. In the folders there is also the original image to be used as comparison to see the results are correct.
2. Extract ORB or SIFT features from the images (SIFT features requires OpenCV>4.4 but typically work a bit better).
3. For each couple of adjacent images in the grid
    a. Compute the match between the different features extracted in (2). For this, OpenCV offers you the `cv::BFMatcher` class. Remember to use L2 distance for SIFT and the Hamming distance for ORB.
    b. Refine the matches found above by selecting the matches with distance less than *ratio * min_distance*, where *ratio* is a user-defined threshold and *min_distance* is the minimum distance found among the matches.
4. You can assume the images are linked together by an affine transform, using the refined matches, find the transformation between the images. To this end, you can use the RANSAC algorithm. The set of inliers can be computed by using the `findHomography()` function, with `CV_RANSAC` as the third parameter (hint: the inliers can be retrieved by using the mask argument).
5. Using the hymnographies found in 4 create the mosaic image merging together the input images according to their relative position.
6. This is the baseline assignment, see the second page for additional suggestions for extra features allowing to improve your mark.

Optional steps

1. You can manually implement a simplified RANSAC following the trace on the slides.
2. You can also propose any algorithm you can figure out different than the proposed one to perform the mosaicing.
3. Try also panoramic images in addition (*not in place of*) to the mosaic:
    a. Project the images on a cylinder surface using the provided static method `cylindricalProj()` of the `PanoramicUtils` class. The method requires as a parameter an angle value (in degrees) which is half of the FoV of the camera used to take the photos. The FoV of the camera is 66° (half FoV=33°) for all the provided datasets excluding the "dolomites" one for which it is 54° (half FoV=27°).
    b. In this case after applying the projection the transformation is a simple translation
4. Acquire your own images (you can take the 9 images moving the camera or acquire a single image, split in 9 parts and try to recombine)
5. Work with color images instead of grayscale
6. Try different feature descriptors
7. Use some blending/mixing technique to better merge adjacent images
8. Equalize the images to avoid color jumps (this can be noticed only in the datasets with illumination change)
9. Try to automatically guess which images are linked to which