Brief Introduction of the approach

The above code is an extension of the code provided in the previous question. Here, the technique of feature matching is modified to use the k-Nearest Neighbours (k-NN) matcher, which is more robust to noise and outliers. The steps followed in the code are as follows:

- Load the images: A list of image filenames is created and the images are loaded using OpenCV's imread function.
- Create a SIFT object: An instance of the SIFT (Scale-Invariant Feature Transform) object is created using OpenCV's cv2.SIFT_create() function.
- Detect keypoints and compute descriptors for each image: The detectAndCompute method of the SIFT object is used to detect keypoints and compute descriptors for each image. The keypoints and descriptors are stored in a list.
- Create a feature matcher: An instance of the BFMatcher (Brute-Force Matcher) object is created using OpenCV's cv2.BFMatcher() function.
- Match keypoints and descriptors between adjacent images using k-NN matcher: The k-NN matcher is used to match descriptors between adjacent images. The knnMatch method of the matcher object is used to find the 2 nearest neighbors for each descriptor in the first image and match them with descriptors in the second image. Only those matches are considered which satisfy the ratio test, i.e., the distance between the first and second nearest neighbor is less than a threshold. The good matches are stored in a list.
- After finding the matches, we estimate the affine transformation between the matched keypoints using RANSAC algorithm. The affine transformation is a linear mapping that preserves the parallelism of lines, and it is used to align the images in a planar panorama.

Special Choices:

In the provided code, the stitching of images is done using OpenCV's stitcher module. The stitcher module provides a high-level API for stitching images together to form a panorama. This module is capable of automatically detecting and aligning overlapping regions between images, and then blending them together seamlessly to create a final panoramic image.

Finally, we stitch the images together using OpenCV's stitcher module, and display the resulting planar panorama.

Difficulties:

Assuming a planar scene, the homography matrix is a transformation matrix that translates points from one picture plane to another. However, getting the ideal homography matrix can be difficult. The fundamental cause is that solving a system of linear equations during the estimation process can be vulnerable to noise, outliers, and degeneracy. Furthermore, the homography matrix depends on the accuracy of the point-to-point correspondence between the two pictures, which can occasionally be challenging to produce. RANSAC (random sample consensus) and its derivatives, which try to

robustly estimate the homography matrix by minimising the effect of outliers and noise, are just one of the strategies that have been developed to solve these problems.

Suggestions:

- Reducing noise and outliers: Noise and outliers have a big impact on the estimation homography matrix's accuracy. Therefore, it is advised to utilise denoising methods to lessen the noise, such as median filtering or Gaussian filtering. Outlier effects can also be reduced by employing outlier rejection techniques like statistical techniques or geometric consistency requirements.
- Handling degeneracy: Another issue that could impair the estimation of the homography matrix is degeneracy. Utilising effective regularisation approaches, such as the Tikhonov regularisation or the total variation regularisation, is crucial for dealing with degeneracy.