Solution and Answers on Homework 4

Solution Explanation:

In this homework we were tasked to implement an augmented reality system using a planar tracking target. Like the previous assignment, we were provided with a code template for the main process and the tracker class with its methods. Additionally, for full credit, we were given a code template for a RANSAC class to implement the RANSAC algorithm.

The process has a predefined procedure for processing the input movie. The program begins by creating a Tracker object, passing it the reference image from which we extract the features to later find in the movie frames, as well as the overlay which we later overlay into the movie.

We then start processing each frame of the input movie. First, we obtain the homography of the frame by using the compute_homography() method of our Tracker object. In the compute_homography() method, we first find matches between the provided frame and reference image by matching SIFT keypoints between the two images. The matches are then filtered using the ratio test. A match is accepted if the first nearest neighbor's distance is less than the ratio threshold times the second nearest neighbor's distance. After that, RANSAC is applied to matches that pass the ratio test, to determine inliers and compute a homography estimate.

After obtaining the Homography for the frame in process.py, we augment that frame with the augment_frame() method. In this method, we draw the overlay image onto the frame. To do this, we first get the warped perspective image of the overlay image using cv2.warpPerspective() and the previously generated homography. We also get a mask of ones and warp it as well. After that, we simply apply the alpha blend and output the augmented frame.

For full credit, we were supposed to implement the RANSAC algorithm using the given RANSAC Class template to find the perfect homography for each frame. We first start by implementing helper functions needed for the RANSAC algorithm to work. First, we need the number of iterations with the compute_num_iter() method with the given formula. Second, we determine the inlier mask using the compute_inlier_mask(), which determines inliers given a homography estimate. A point correspondence is an inlier if its re-projection error is below the given threshold. The last step is just applying the RANSAC algorithm discussed in class. How it is done is best observed in the code. Note the comments in the code and the code itself for further clarification and the corresponding return types.

Qualitative Evaluation Questions:

- Q: Once you have the system implemented and working, inspect the results and in your report, describe qualitatively how well the system works. When does it produce accurate results and when does it falter?
    - A: The system works best with a clear view of the reference image in the frame as well when we have a straight perspective from up top on the reference image (SIFT features cannot really get recognized at an extreme angle). For example, having not enough features due to obstruction of the view of the image in the frame leads to a distorted overlay.