**Written Assignment 2**

1. Using “Rainier1.png” and “Rainier2.png” run RANSAC using a small number of iterations. How many iterations are necessary to reliably find the correct homography? What percentage of the matches are inliers? If only 5% of the matches were inliers, how many iterations of RANSAC would need to be run?

For these two photos, I’ve found the optimal number of inliers for the photo is approximately 338, from 643 matches (53%), when running with 9999 iterations. When using fewer iterations, hwoever, I’ve found that 50 iterations still produces reliable results (9/10 times, it produces >300 inliers). 25 iterations yielded less reliable results, only about 60-75% of the time yielding >300 inlier counts.

5% of 643 matches is 32 inliers. If only 32 inliers are desired, using 10-15 iterations of RANSAC is very likely to produce at least that many inliers. However, it is unlikely for the pictures to stitch accurately with such a lower number of inliers, so it’s unlikely to be worthwhile to run fewer RANSAC iterations to this level.

2. Look at the function ComputeDescriptors. Is this descriptor invariant to intensity offset and intensity gain differences? Is it invariant to spatial translation, scale or rotation? If not, why?

ComputeDescriptors looks at eight of its neighbors, each 4 pixels away along the horizontal or vertical axes (for instance, (-4,-4) away, (-4,0) away, (-4,4) away, etc). For each of those neighbors, it stores the difference between that neighbor’s green color channel and its own green color channel. These eight differences represent the descriptor for each interest point.

Given that the difference between the two color channels are stored, this descriptor should be invariant to intensity offset/gain differences. Additionally, it should be invariant to translations as well, because the neighbors are chosen by looking relatively to the center of the interest point. However, because the neighbor is chosen by a constant factor, this means it is not invariant to scale changes. Similarly, the neighbors are always observed in the same directions, and no attempt is made to rotate the descriptor into a common orientation, so it is not invariant to rotation.

3. How would vignetting make image stitching more difficult? What artifacts might you see?

Vignetting reduces the brightness as the distance from the center of the image increases. This can complicate stitching, because the same object may appear in two objects with different brightnesses, causing problems with matching interest points across the two images.

Artifacts that may be seen in an image stitched from vignetted photos include dark spots near seams, and/or poor seams due to trouble matching consistently.

4. If we wanted to stitch two images like “Hanging1.png” and “Hanging2.png” that are rotated relative to each other, how would the code need to be updated? What functions would you change?

ComputeDescriptors would need to be updated to determine a descriptor that is invariant to rotations (the DESC\_SIZE constant would also need to be updated, in the event that more size is needed for the descriptor). A very simple modification that could be made to enable this is to rotate the descriptor so that the brightest part of the color channel is always facing “up”.

No other functions should need to be changed. ComputeHomography is what determines the final projection, but it doesn’t receive any of the descriptors. MatchInterestPoints is the only function that does read the descriptors, but assuming it is correctly using DESC\_SIZE, it shouldn’t need to know anything specific about the descriptor itself.

5. How would radial distortions (Szeliski, Section 2.1.6) of an image affect a panorama stitcher? What code would need to be changed?

Radial distortions turn what should be seen as straight lines into curved lines. Common types of radial distortions include barrel/fisheye distortions (straight lines bending outward from the center) and pincushion distortions (straight lines bending inward). These distortions would make it difficult for a panorama stitcher, as it expects pictures to be similar, but for a barrel-distorted photo, the right side of one photo would have straight lines bending to the right, while the left side of a second photo might have the same object’s lines bending to the left.

Images would need to have the distortion removed prior to stitching, typically by applying the reverse distortion to undo the first one (pinching things inward by applying a pincushion effect to a barrel-distorted image).

6. Look at the image “AllStitched.png”. Why are the images on the end stretched and distorted?

When someone wants to take photos for a panorama, they take multiple photos looking in different directions from the same vantage point. As such, projecting the photos straight onto a flat surface yield strong distortion near the sides of the photos (the outer angles spread out much faster on the projected flat surface than the angles closer to 0°).

To produce a more natural representation, it would be better to project the panorama against the inside of a cylindrical shape, and then unwrap that shape onto a flat surface. This would help with the outer-edge distortion.