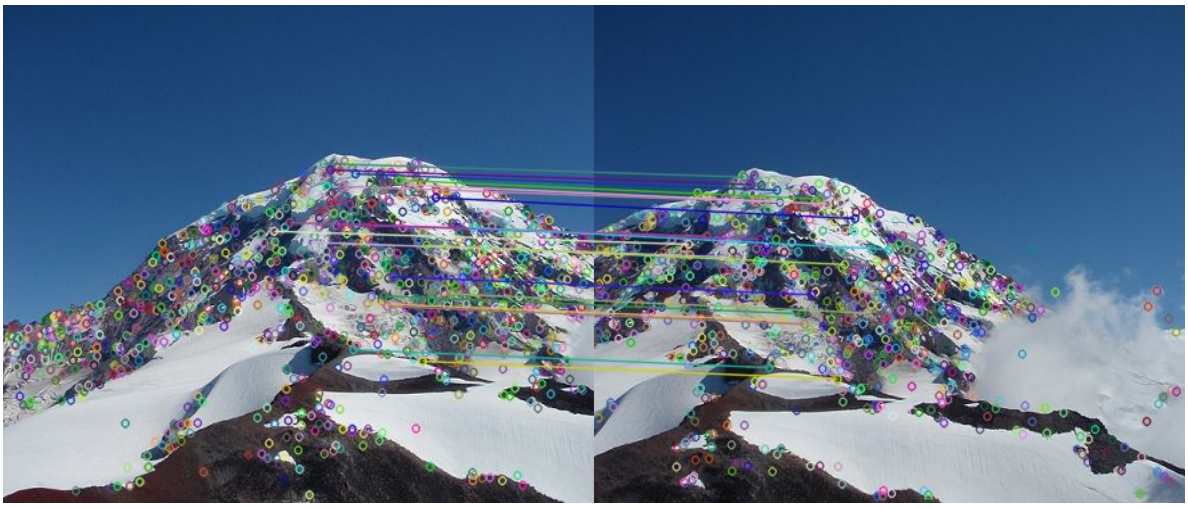
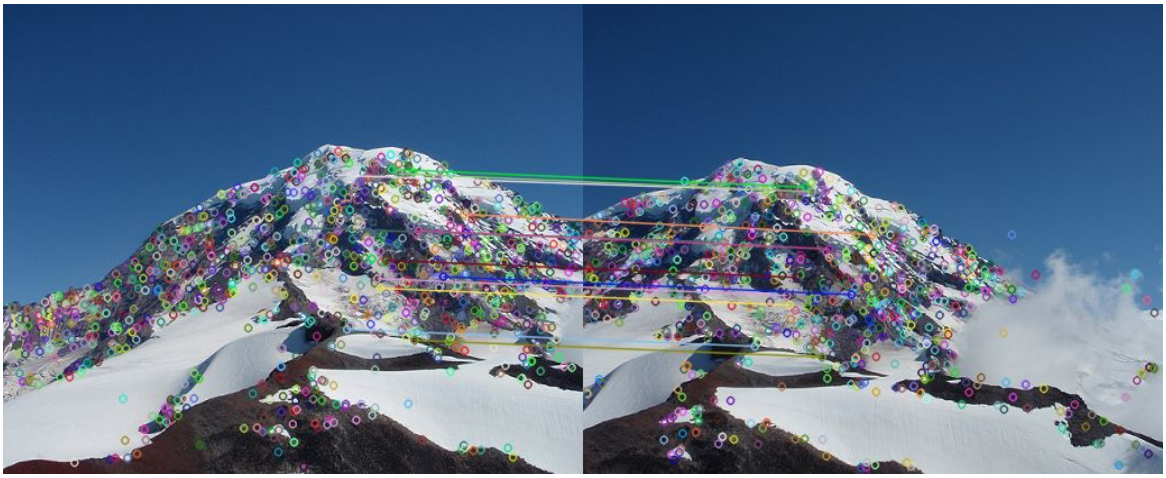
- **Part1**
  - In this part, I used Harris corner detection algorithm from Assignment 2
  - The corner detection applied on ‘Boxes.png’, ‘Rainier1.png’ and ‘Painier2.png’ and saved the result to ‘results’ folder.

-
- **Part2**
  - Using OpenCV SIFT implementation, I found keypoints and descriptors of ‘Rainier1.png’ and ‘Rainier2.png’
  - I used the matching function implemented in assignment2 to match descriptors of two images.



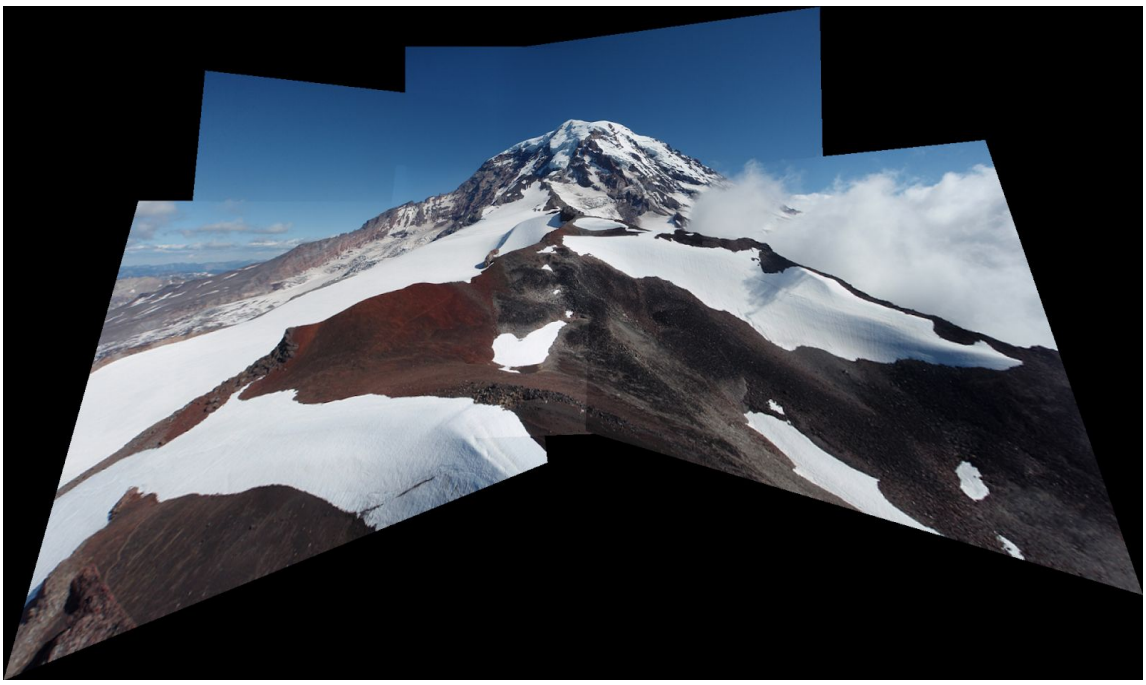
- **Part3**
  - For this part, I implemented ‘project’, ‘computeInlierCount’ and ‘RANSAC’ as instructed in the project description.

- I found inliers of Part2 results using the RANSAC function.



- **Part4**
  - In this part, I implemented ‘stitch’ and ‘calc_stitch_dims’.
  - ‘Calc_stitch_dims’ finds the dimensions of the result of stitching two images.
  - ‘Stitch’ stitches two images as instructed in the project description.
- **Extra Credit 1**
  - In this part, I implemented two classes. The first class is ‘node’, which represents an image with its keypoints and descriptor.
  - Second class in ‘match_set’ which gets two node objects as an input and calculates their matched keypoints. ‘Match_set’ object is also able to perform RANSAC and stitch to the pair. I used the previously implemented RANSAC and stitch functions with minor changes for this object.
  - Finally, I implemented the ‘stitch_all’ function which takes an image set and stitches all images together. To stitch all images, in each step algorithm finds the closest match to the currently stitched image. To get the ball rolling, I assigned the first image to be the initial stitched image. After matching each image, the

algorithm removes it from the image set.



- **Extra Credit 2**

- For this part, I used an image set I took from a pergola in my neighborhood park.

- **Extra Credit 4**

- For blending, I used the center weighting algorithm. For each overlapped pixel, I weight pixels from image1 and image2 by their distance to the center of that image.