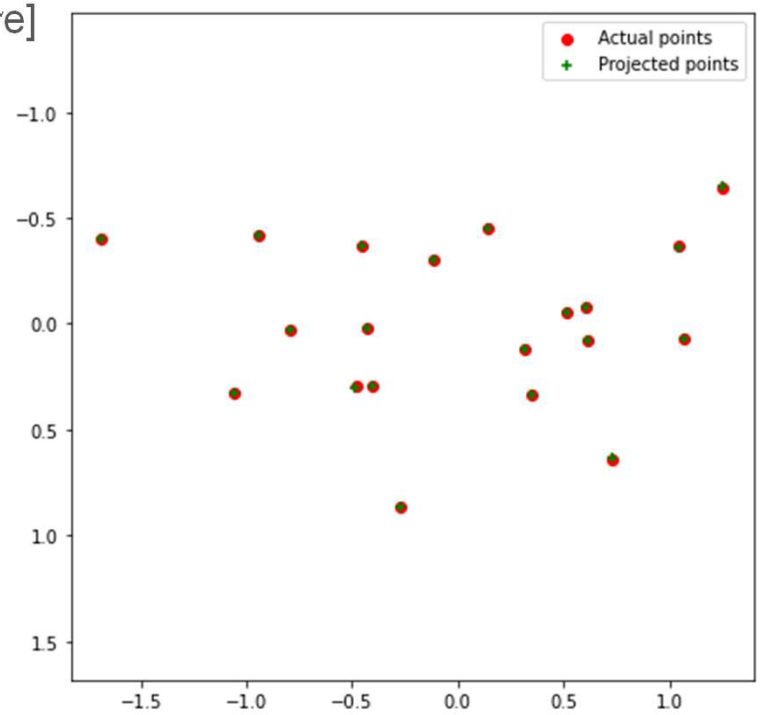
CS 4476/6476 Project 3

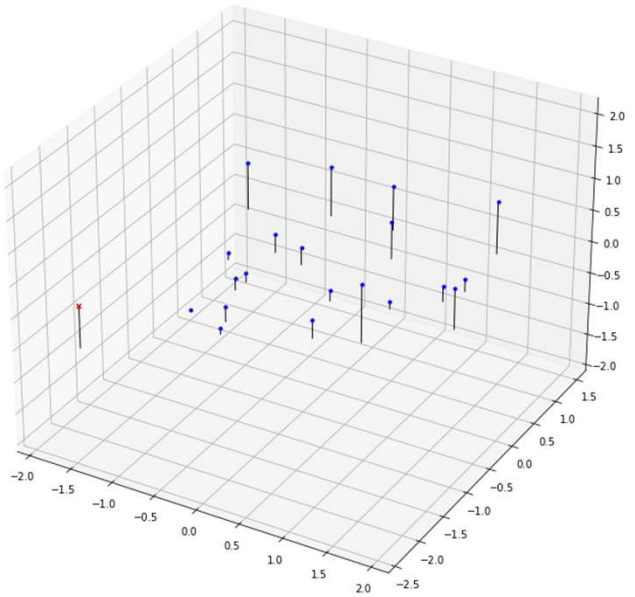
Cameron Potter
cpotter8@gatech.edu
cpotter8
903465425

Part 1: Projection matrix

[insert visualization of projected 3D points and actual 2D points for the CCB image we provided here]

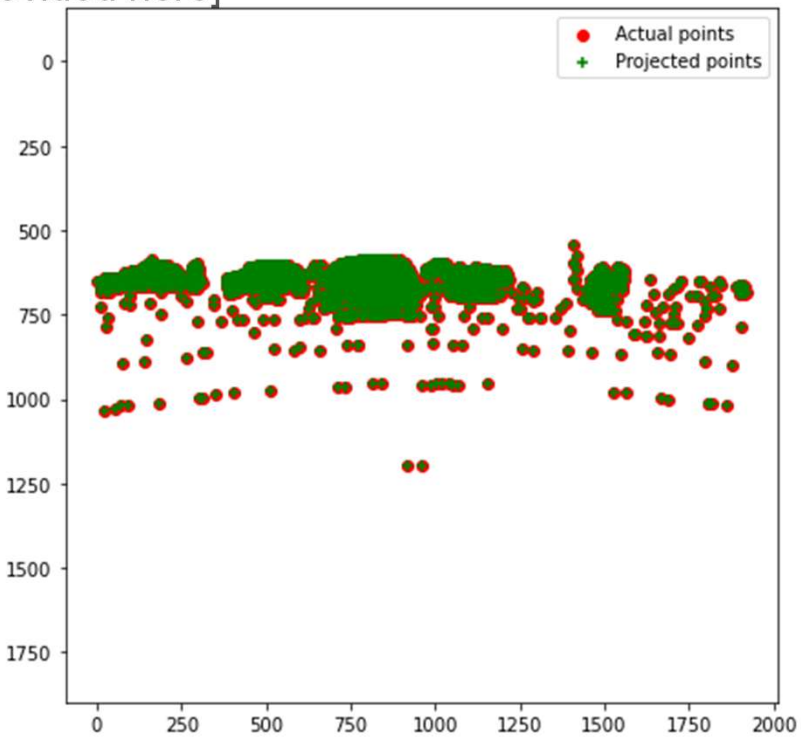


[insert visualization of camera center for the CCB image here]

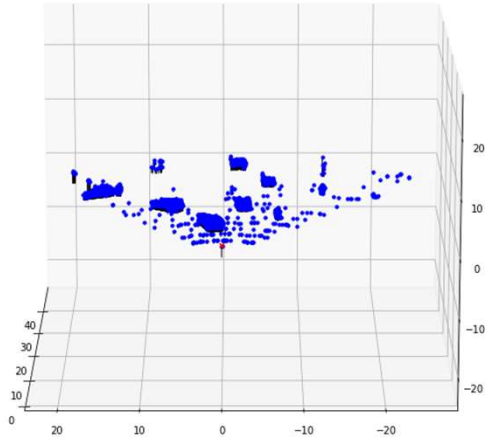


Part 1: Projection matrix

[insert visualization of projected 3D points and actual 2D points for the Argoverse image we provided here]



[insert visualization of camera center for the Argoverse image here]



Part 1: Projection matrix

[What two quantities does the camera matrix relate?]

It relates world 3D coordinates and 2D image coordinates.

[What quantities can the camera matrix be decomposed into?]

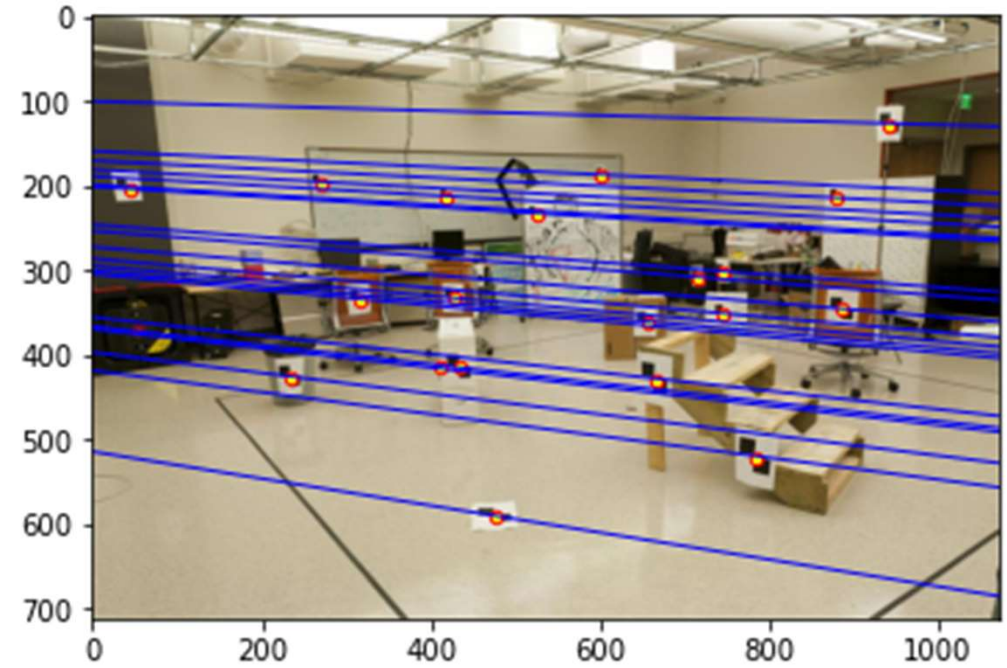
intrinsic and extrinsic parameters

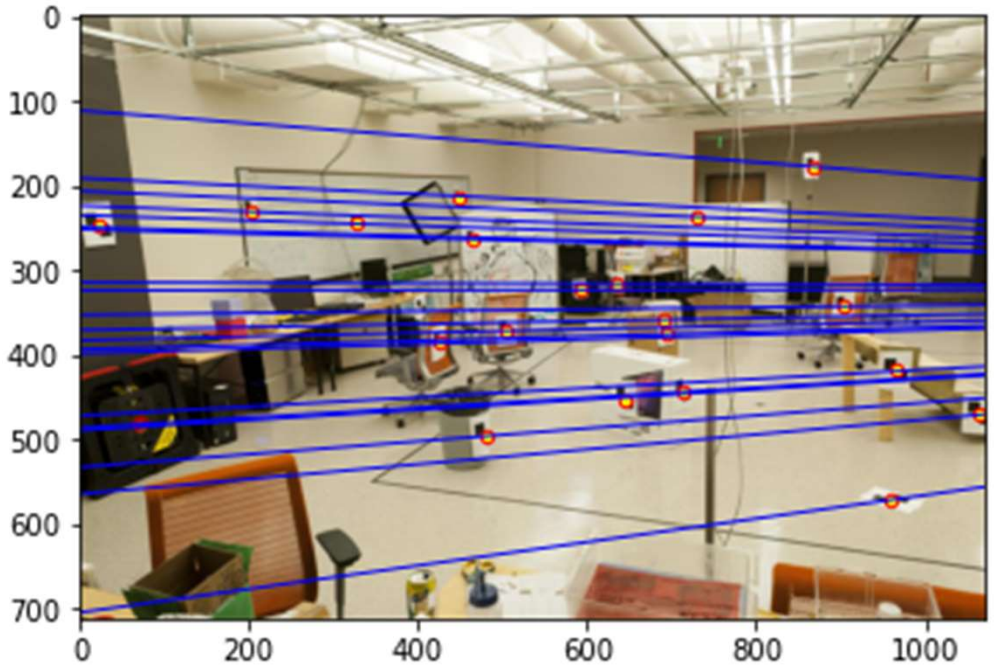
[List any 3 factors that affect the camera projection matrix.]

Rotation, translation, and camera position

Part 2: Fundamental matrix

[insert visualization of epipolar lines on the CCB image pair]





Part 2: Fundamental matrix

[Why is it that points in one image are projected by the fundamental matrix onto epipolar lines in the other image?]

The points in the images are a camera projection matrix times the real world coordinates. We can derive the fundamental matrix using linear regression tequniques on a set of points, allowing for the points to be projected to the other points' epipolar lines.

[What happens to the epipoles and epipolar lines when you take two images where the camera centers are within the images? Why?]

The epipolar lines would intersect in the epipolar plane, and the epopiles would switch relative direction (if we have two epipoles, A and B, and A', B' are the epipoles with the camera centers are within the images, and WLOG the x coordinates of them are Ax and Bx respectively, then Ax-Bx = -(A1x – B1x)). This is because the two planes created by the two images would then intersect inside the epipolar plane.

Part 2: Fundamental matrix

[What does it mean when your epipolar lines are all horizontal across the two images?]

Each point's corresponding point will be along the horizontal epipolar lines.

[Why is the fundamental matrix defined up to a scale?

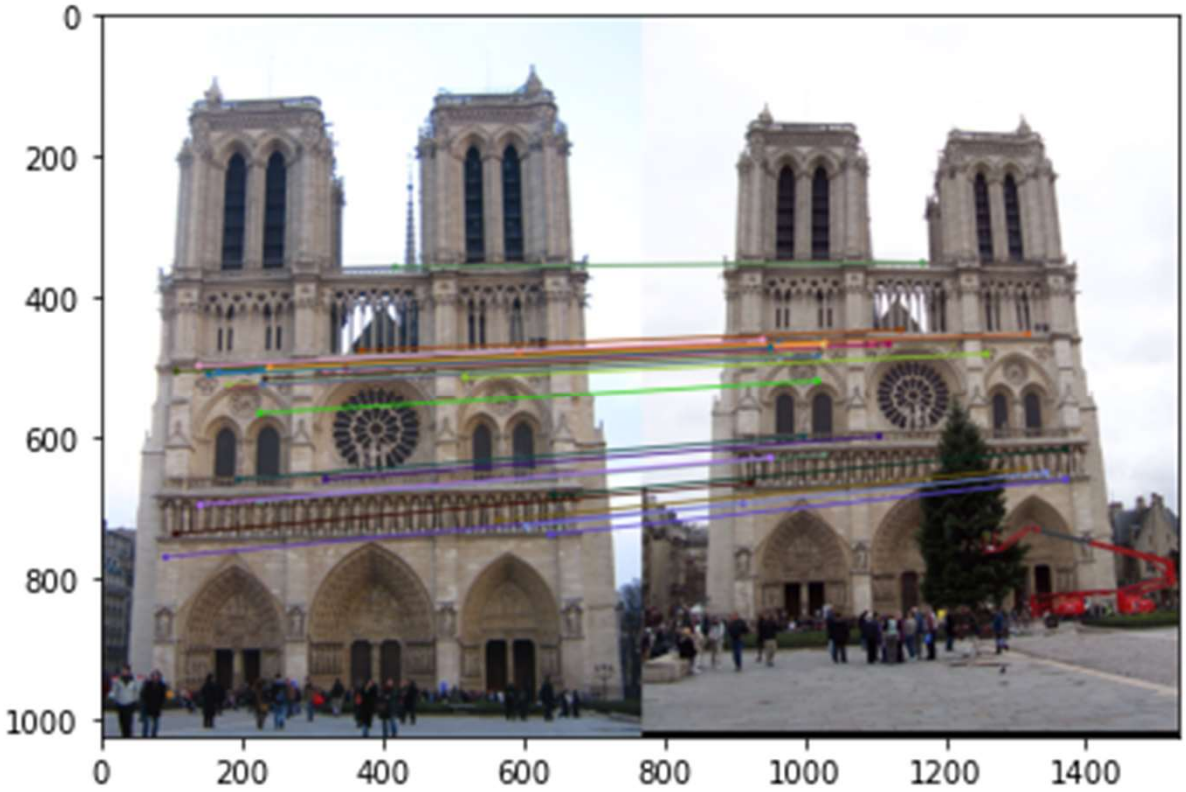
You can multiply the formula to get the matrix my any scalar and the new matrix will also be a valid fundamental matrix

[Why is the fundamental matrix rank 2?]

Epipoles themselves must lie on epipolar lines, hence the rank 2.

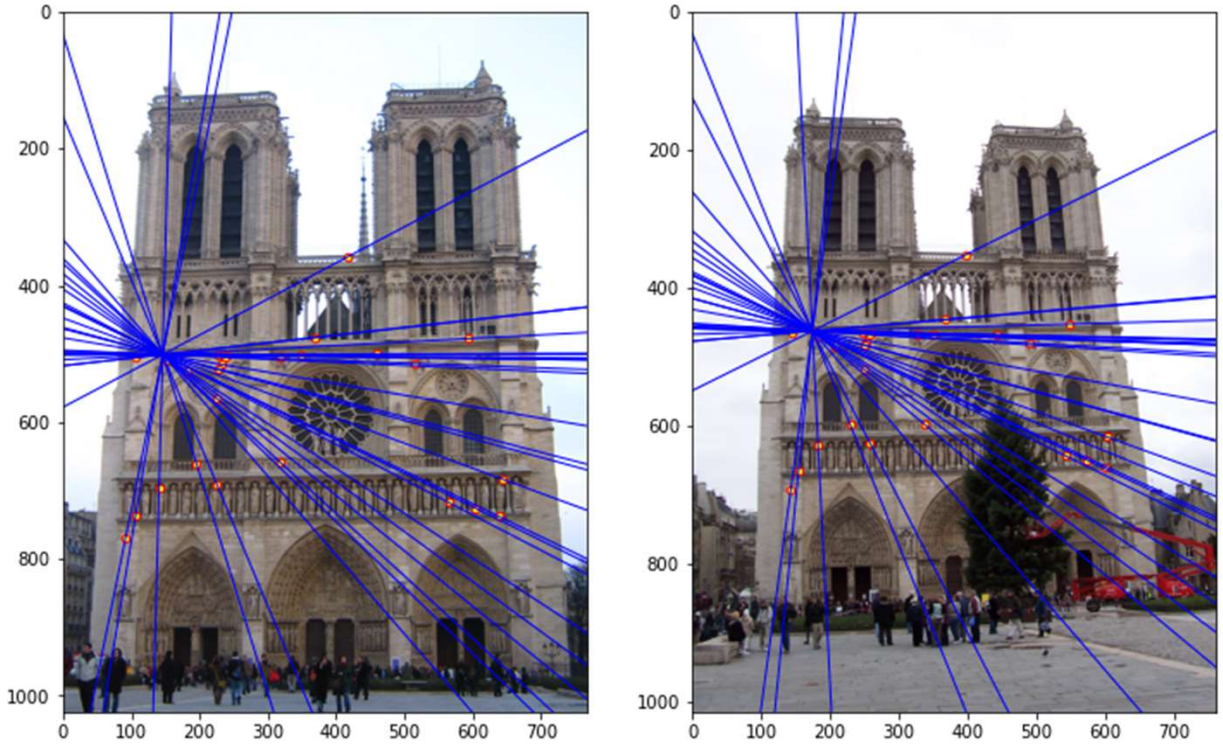
Part 3: RANSAC

[insert visualization of correspondences on Notre Dame after RANSAC]



Part 3: RANSAC

[insert visualization of epipolar lines on the Notre Dame image pair]



Part 3: RANSAC

[How many RANSAC iterations would we need to find the fundamental matrix with 99.9% certainty from your Mt. Rushmore and Notre Dame SIFT results assuming that they had a 90% point correspondence accuracy?]

calculate_num_ransac_iterations(0.999, 8, 0.9) = 12

[One might imagine that if we had more than 9 point correspondences, it would be better to use more of them to solve for the fundamental matrix. Investigate this by finding the # of RANSAC iterations you would need to run with 18 points.]

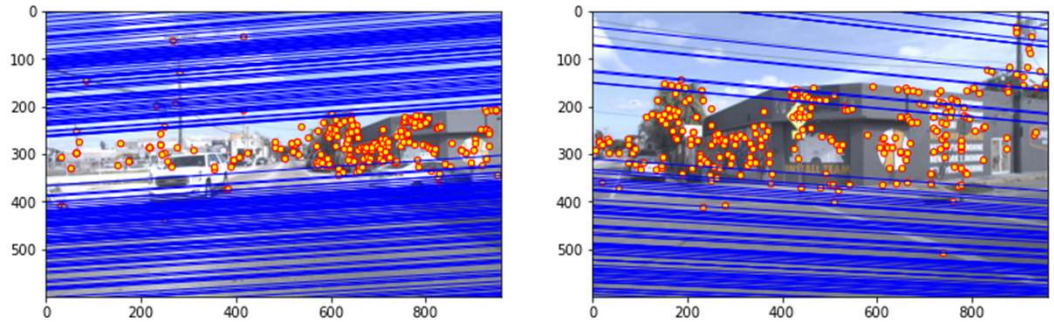
calculate_num_ransac_iterations(0.999, 18, 0.9) = 42

[If our dataset had a lower point correspondence accuracy, say 70%, what is the minimum # of iterations needed to find the fundamental matrix with 99.9% certainty?]

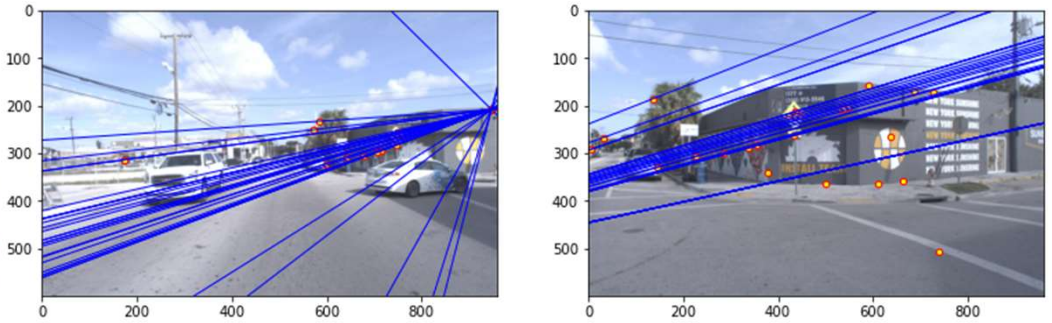
calculate_num_ransac_iterations(0.999, 8, 0.7) = 116

Part 4: Performance comparison

[insert visualization of epipolar lines on the Argoverse image pair using the linear method]



[insert visualization of epipolar lines on the Argoverse image pair using RANSAC]



Part 4: Performance comparison

[Describe the different performance of the two methods.]

The correspondences look very similar, but the RANSAC implementation seems to have slightly better accuracy.

[Why do these differences appear?]

Without RANSAC, we are simply estimating the fundamental matrix, which includes outlier points. The RANSAC implementation helps remove the outlier points to get a better fit.

[Which one should be more robust in real applications? Why?]

In real applications, there will certainly be outliers in data. Hence, RANSAC will be more robust to better fit noisy data.

Part 5: Visual odometry

[How can we use our code from part 2 and part 3 to determine the “ego-motion” of a camera attached to a robot (i.e., motion of the robot)?]

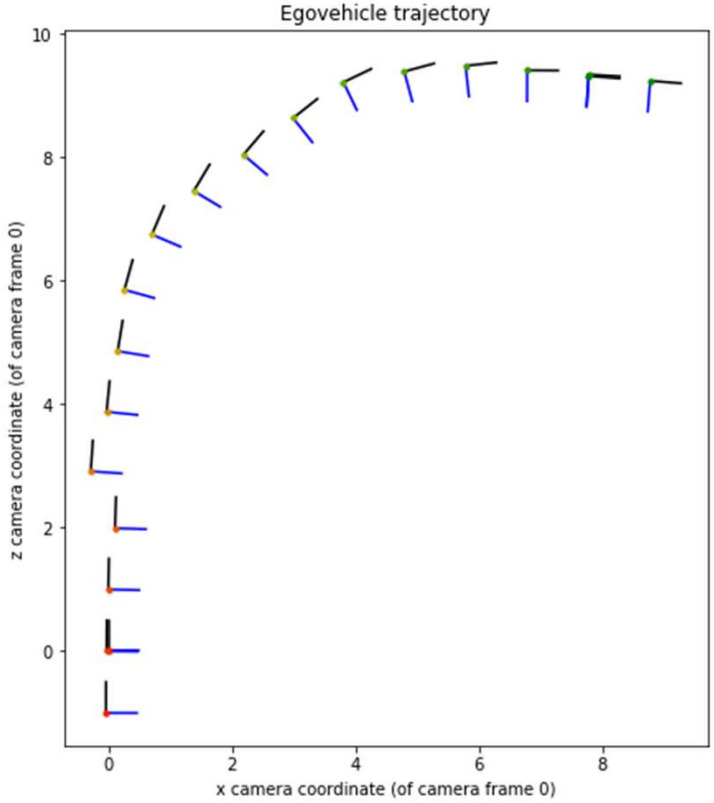
We can use the fundamental matrix and RANSAC to get the rotation and translation of the real-world positions between each frame. This allows us to get the “ego-motion” of a camera attached to a robot, as we can see how the camera moves over time using the point correspondences.

[In addition to the fundamental matrix, what additional camera information is required to recover the ego-motion?]

We need the essential matrix, camera parameters, relative camera rotation and translation, and previous world frame pose

Part 5: Visual odometry

[Attach a plot of the camera's trajectory through time]



Part 6: Panorama Stitching

[Please add a README style documentation here for your implementation of panorama stitching with: description of what you implemented, instructions on how to replicate the results in clear steps that can be followed by course staff. Failure to replicate results by following this documentation will result in point penalties on this question of the assignment.]

Here, we implement panorama stitching using SIFT, homography matrix, and a warp operation with OpenCV. First, we load in our images and get grayscaled versions to feed into SIFT, returning interest points. We then use BFMatcher's knnMatch to get initial matches. We get our final candidate matches, we use a threshold multiple of 0.5 between the first and second distances returned in the previous step (that is, half the second distance must be greater than the first distance). We then get our source and destination from the SIFT returns into linearized homogeneous arrays, which we input into cv.findHomography along with RANSAC as our method choice. Next, we multiply by the homography matrix to get the transform, and then divide by the last coordinate to get Cartesian coordinates. Finally, we utilize cv.remap to perform the interpolation. We can input the imageB into the beginning of the returned image to get our panorama.

Part 6: Panorama Stitching

[Insert visualizations of your stitched panorama here along with the 2 images you used to stitch this panorama (**there should be 3 images in this slide**)].