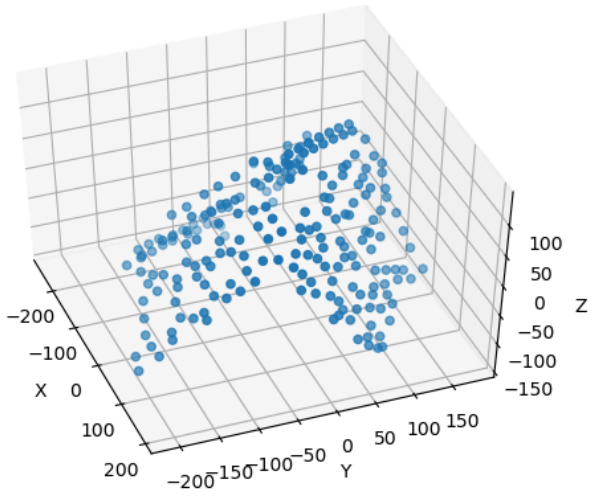
CS543/ECE549 Assignment 5

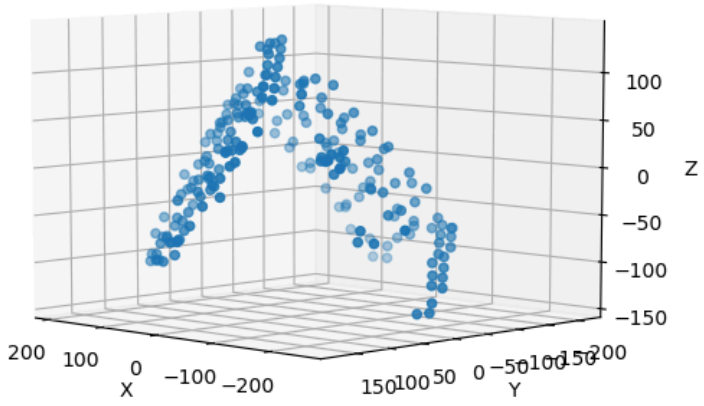
Name: Karan Pandya
NetId: karandp2

Part 1: Affine factorization

A: Display the 3D structure (you may want to include snapshots from several viewpoints to show the structure clearly). Report the Q matrix you found to eliminate the affine ambiguity. Discuss whether or not the reconstruction has an ambiguity.

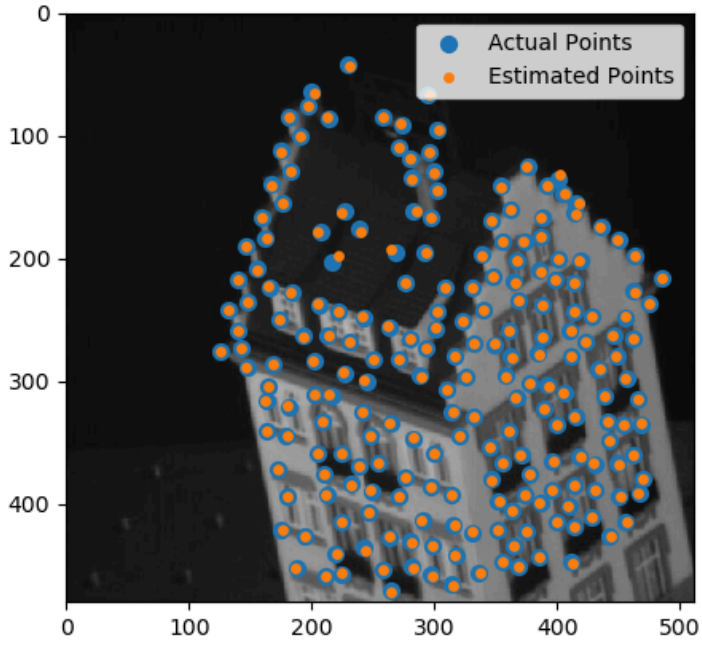
Q is [[0.080 0. 0.]
[0.0001 0.085 0.]
[0.0027 0.0059 0.043]]



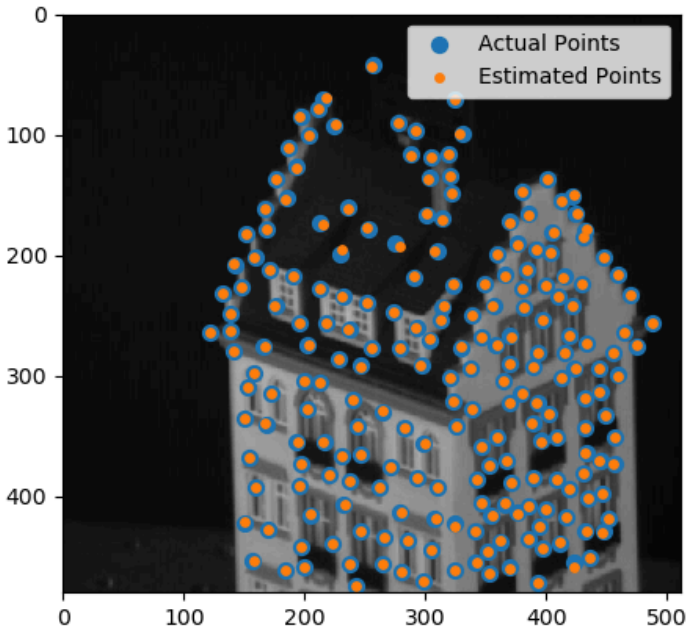


Yes, there will be still ambiguity after eliminating affine ambiguity. There are several other factors which cause ambiguity like measurement noise, lack iof correspondence,lack of accurate camera calibration, dynamic scene etc.

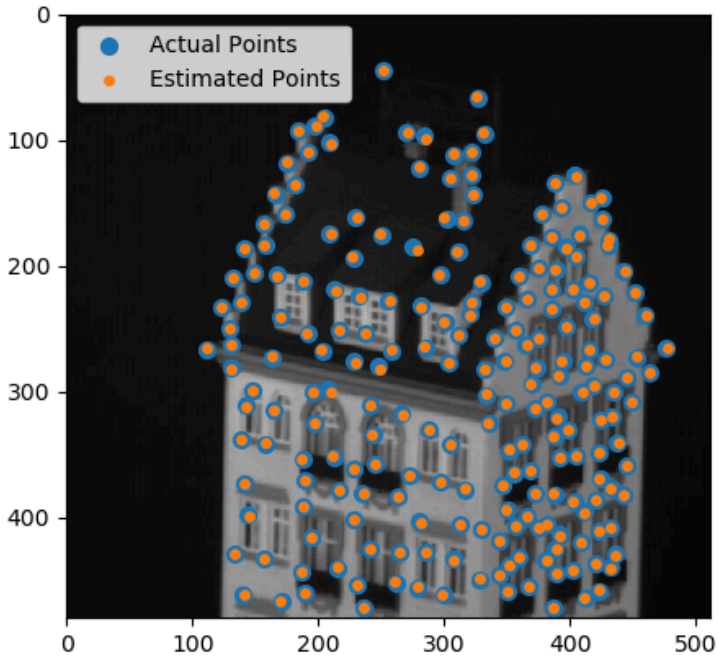
B: Display three frames with both the observed feature points and the estimated projected 3D points overlayed.



Frame number = 25



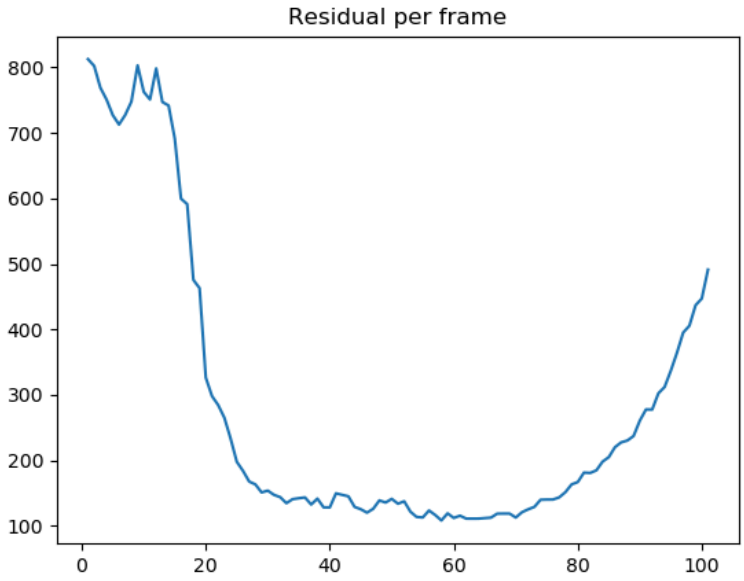
Frame number = 50



Frame Number = 75

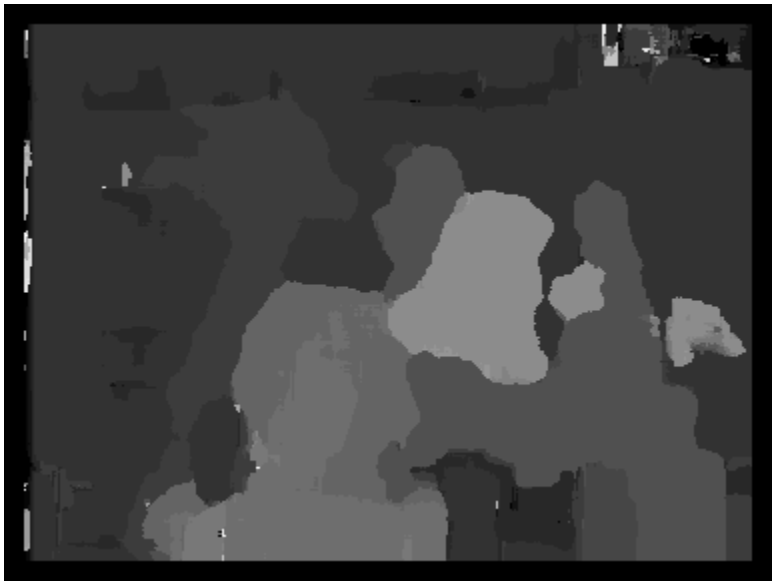
C: Report your total residual (sum of squared Euclidean distances, in pixels, between the observed and the reprojected features) over all the frames, and plot the per-frame residual as a function of the frame number.

Total Residual for all frames: 28586



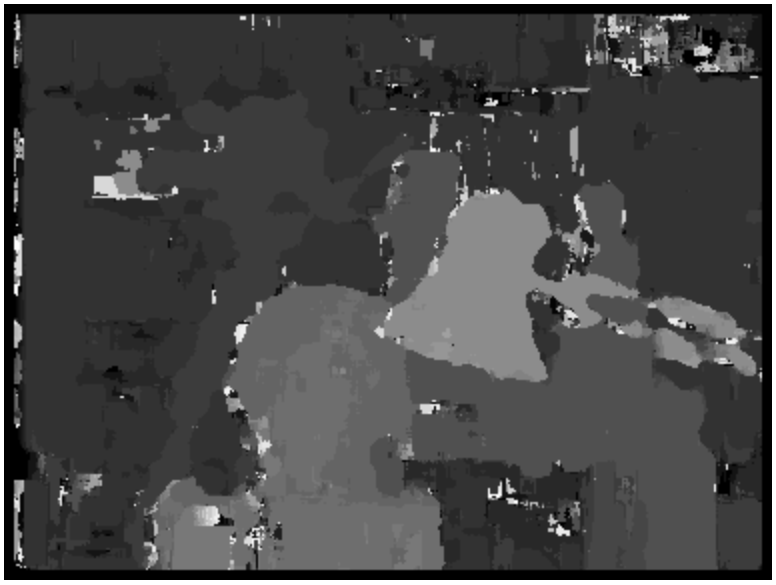
Part 2: Binocular stereo

A: Display best output disparity maps for both pairs.



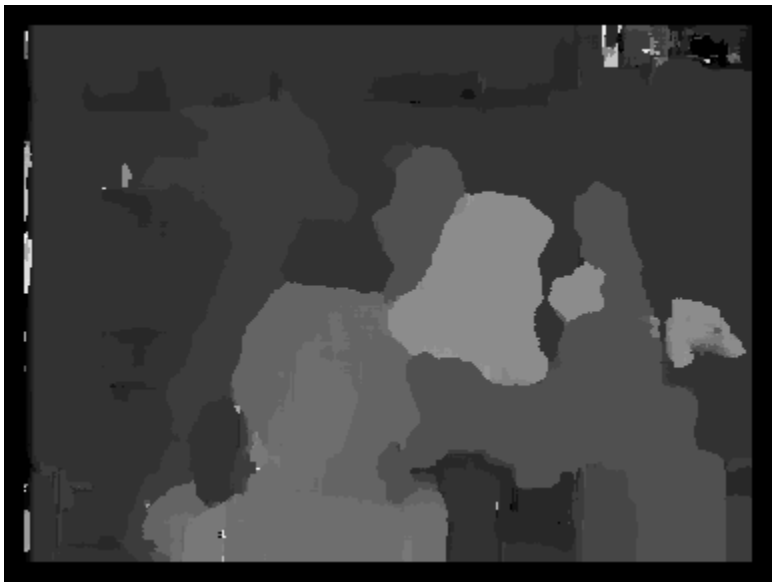
B: Study of implementation parameters:

1. **Search window size:** show disparity maps for several window sizes and discuss which window size works the best (or what are the tradeoffs between using different window sizes). How does the running time depend on window size?



Window size =5

RunTime = 15.3s



Window size = 10

RunTime = 16.4s

Window size = 15

RunTime = 18.2 s

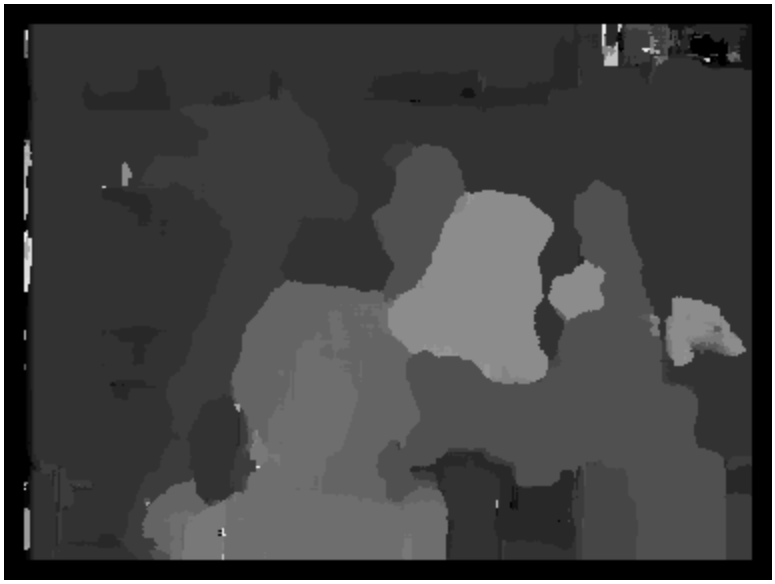
Window size 10 works the best as it is a balanced tradeoff between the smoothness of depth map vs the noise in depth map.

Smaller windows offer more detail but introduce noise, while larger ones create a smoother but potentially less accurate depth map due to lost image details. Additionally, increasing window size escalates computation time as it processes more information for each image pixel.

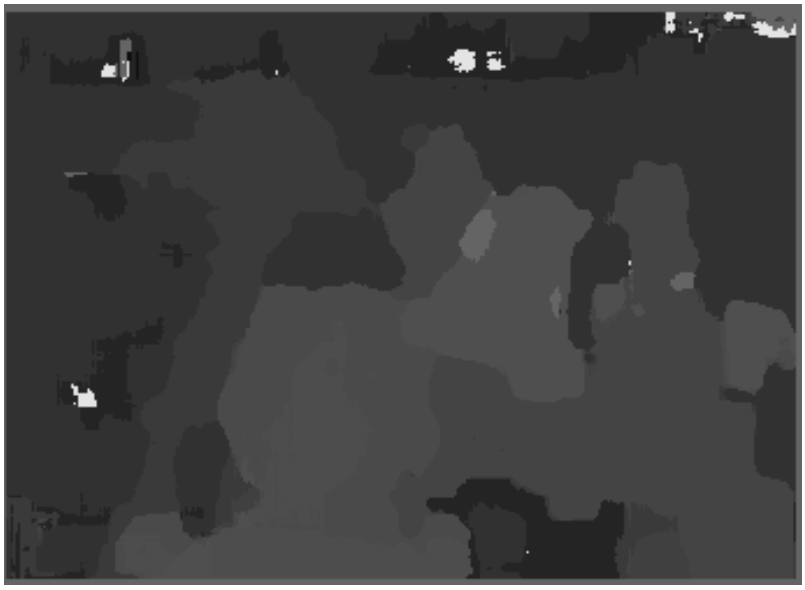
2. **Disparity range:** what is the range of the scanline in the second image that should be traversed in order to find a match for a given location in the first image? Examine the stereo pair to determine what is the maximum disparity value that makes sense, where to start the search on the scanline, and which direction to search in. Report which settings you ended up using.

   Determining the range of scanlines necessary to find a match in the second image varies and often involves trial and error. For instance, in the Tsukuba stereo pair, the maximum disparity is approximately 12 pixels, while in the Moebius image pair, it's around 40 pixels. The search typically starts from the same pixel as the left image on the second image and spans equally in both left and right directions, respecting the maximum disparity of the images.

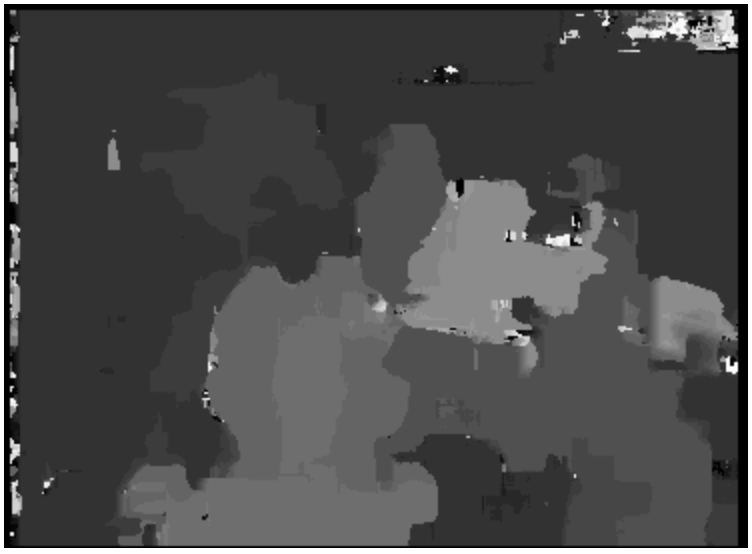
   Matching function: try sum of squared differences (SSD), sum of absolute differences (SAD), and normalized correlation. Show the output disparity maps for each. Discuss whether there is any difference between using these functions, both in terms of quality of the results and in terms of running time.



SSD = 16.4s



SAD = 17.8s



NCC = 46.2s

SSD demonstrates the highest performance, followed by SAD, and then NCC. However, it's worth noting that while SSD and SAD computations are relatively faster, NCC requires significantly more time to compute due to its complexity.

C: Discuss the shortcomings of your algorithm. Where do the estimated disparity maps look good, and where do they look bad? What would be required to produce better results? Also discuss the running time of your approach and what might be needed to make stereo run faster.

The apparent shortcomings of the algorithm are that the edges of objects are not as accurate as they should be and there is also lot of noise in some parts of the image. The estimated disparity maps look good for objects closer to the camera but look bad for objects farther away since it”s harder to estimate disparity for far away objects as the disparity reduces.

To make the stereo run faster one approach could be to use dynamic programming by determining the best matching pairs by considering the cost or similarity between pixels in the left and right images across different disparities. This reduces computational complexity and results in faster runtimes.

Part 3: Extra Credit

Find additional rectified stereo pairs on the Web and show the results of your algorithm on these pairs.
Input images from web:

SSD disparity map: