ADVANCED SIGNAL PROCESSING - ASSIGNMENT 4

Kommineni Aditya - EE17B047

Structure from Motion

IMPLEMENTATION DETAILS :

- Firstly, we calculated the normalized camera coordinates using the intrinsic camera matrix and the matched points matrix.
- Then, we calculated the reprojection error for back projecting the reference array 3D points onto the various views.
- Here, we can wary the number of views we use for reprojection. In this case, we wary the views among 5, 15, 25.
- Now, we can minimize the reprojection error using optimization techniques such as least squares in scipy.
- In order to facilitate the minimization, we use W(the depth array for each matched point), theta(the angles which the various views make with respect to the reference view) and T(the translation vectors which refer to the displacement of the reference view with respect to the other views) as the dependent variables and minimize the residues which the reprojected points make with the original matched 2D points.
- I found that when running the optimization to minimize the sum of residuals, it was providing all the depths to be equal whereas when optimizing individual residuals, it gave the optimal output.
- Once we perform the optimization, we have the required W, theta and T matrices. Then, we compute the 3D points for each matched point with respect to the reference frame using the formula below :

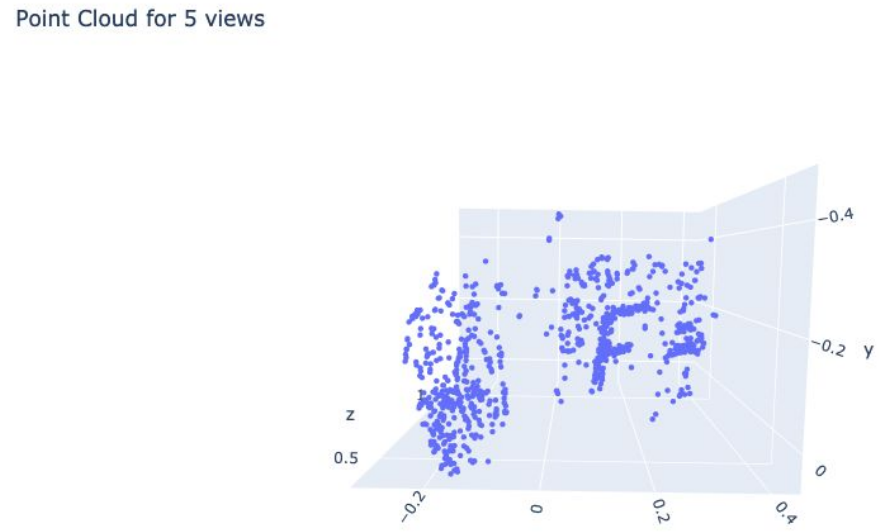
$$X = Rx/W + T$$

  In the formula above, x corresponds to the 2D homogeneous coordinates.

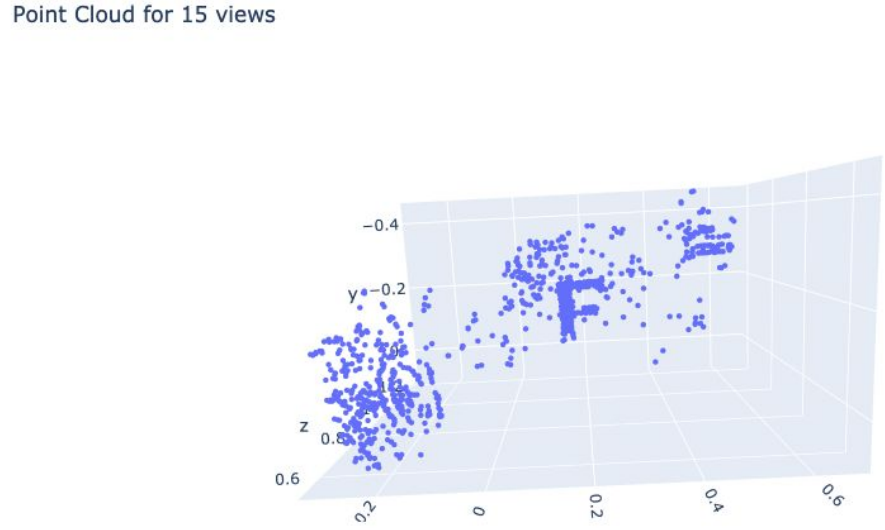
- Now, we can plot the point cloud of 3D coordinates in a scatter plot. For this, we have employed the plotly package.

PLOTS :

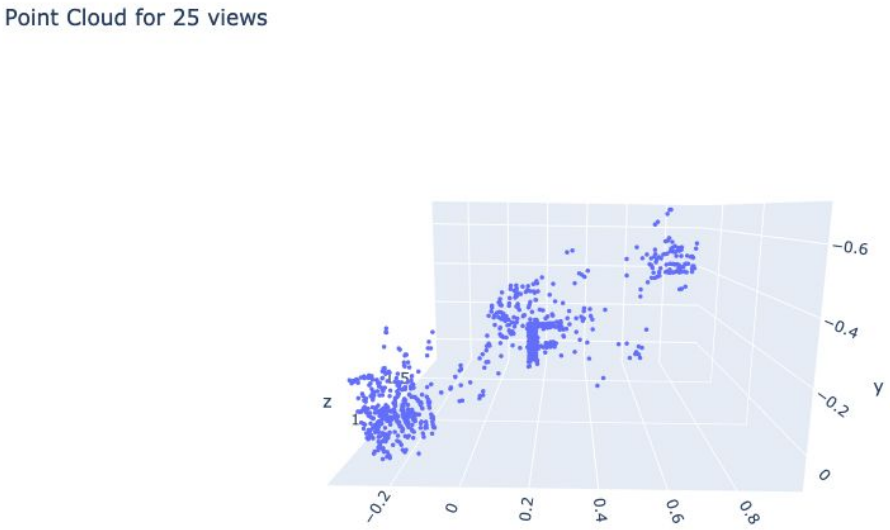
- The plot for point cloud with 5 views is as shown below :



- The plot for point cloud with 15 views is as shown below :



- The plot for point cloud with 25 views is as shown below :



OBSERVATIONS :

- From the point cloud plots above, we can see that as we increase the number of views, the point clouds tend to better represent the depth of the matched points.

- During running the code, when running least squares, the ability to reduce the error with respect to all the residues of the reprojected points gives a better optimal solution over the case wherein we use the sum of the difference as the error.

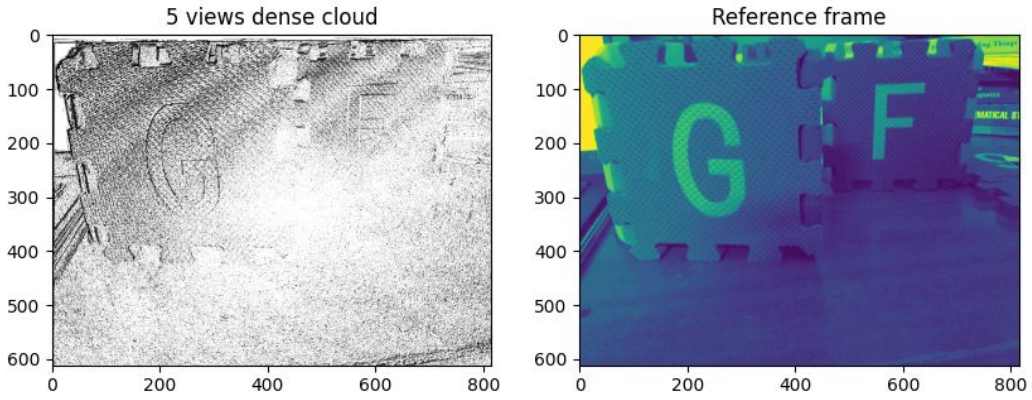
Plane Sweep

IMPLEMENTATION DETAILS :

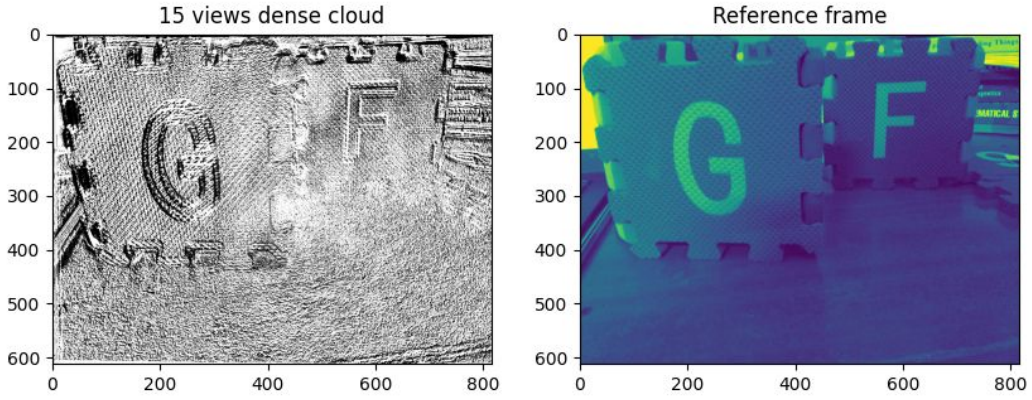
- From the previous SFM, we obtained the R, T, W matrices for the views of the image.
- In order to obtain a depth map, firstly we take about 10 depth values in between the maximum and minimum depth obtained from the W array.
- Then, we find the homographies for reference view and other views and for each depth plane from the values used above.
- We compute the homographies using cv2.warpPerspective function. I did try using the method as done in Assignment3. However, it was slow. Hence, I resorted to using the openCV function.
- Once we compute the homographies, we can compute the projection of the images in each perspective onto the reference frame. Then, we can calculate the variance and determine the depth at which for each pixel the variance is the least. Using this, we assign a depth to each pixel of the image.
- Then, we can plot the depth map of the reference frame image.

PLOTS :

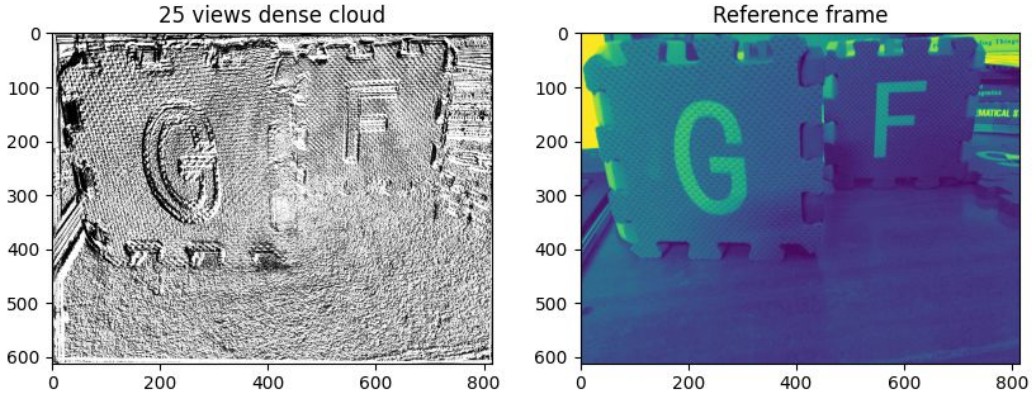
- The plot for the dense depth map for 5 views is as shown below :



- The plot for the dense depth map for 15 views is as shown below :



- The plot for the dense depth map for 25 views is as shown below :



OBSERVATIONS :

- From the plots above, we can see that as the number of views for the images increases, the details of the depth image increases i.e. in the case of 5 views, the depth information is almost absent and other features in the images are as well not present. Whereas in the depth map for 15 and 25 views, the details of the image are better captured and the depth information as well is better viewed.