CS4277 Lab 4 Report

Nicholas Sun Jun Yang A0217609B

get_plane_sweep_homographies

1) Get the Rotation and Translation matrix from the Camera pose

2) Set the unit length normal of the plane, n_m as [0 0 1] for a Fronto Parallel sweep

3) Compute the inversed Camera matrix, K_ref_inv

4) For every depth, set the homography matrix relating the 2 planes to be np.dot(K, np.dot(R_k + d_m_inv * np.dot(C_k, n_m.T) , K_ref_inv))

compute_plane_sweep_volume

1) With respect to the reference image, which from the lab instructions has the reference id of 4, calculate the relative pose of each image to the reference image.

2) For every depth, with respect to the reference image, calculate the homography of each image and then warp each iamge to the reference frame

3) Get the total number of pixels mapped successfully and accumulate the accum_count based on them

4) Compute the absolute pixel wise intensity differences between the reference image and the warped image

5) Compute the average intensity difference across all color channels for each pixel to get the variance

6) Ignore the pixels that were not mapped and set their variance to none

7) Update the plane sweeping volume with the variance

compute_depths

1) For every pixel in the plane sweep volume,extract the volume variances across different depth values and find the index corresponding to the minimum variance

2) Retrieve the inverse depth corresponding to the index with minimum variance and assign it to the corresponding pixel in the inverse depth image.

unproject_depth_map

1) First get a grid of the image's pixels' x,y coordinates as well as the focal length coordinates and the camera's principal point from the Camera Instrinsic Matrix K

2)Center the x coordinates by the principal point and then scale the x coordinates by the depth to convert it to a 3D point. Next scale the x coordinates by the focal length to convert them to the world frame. Do the same thing for the y coordinates.

3) For the case when there is a mask, apply the mask on the x and y coordinates.

5) Reshape the image to get the RGB values for the points

6) Stack up the x,y coordinates with the depth values and reshape them to get the 3D coordinates of the points

post_process

1) Compute the inverse depth map from the plane sweep volume and the depths

2) Create the mask by only considering pixels that are within the range of np.mean(

inv_depth_image) + (2.5 * np.std(inv_depth_image))

3) Denoise the inverse depth image using the scipy library's gaussian filter function

Results are on next page

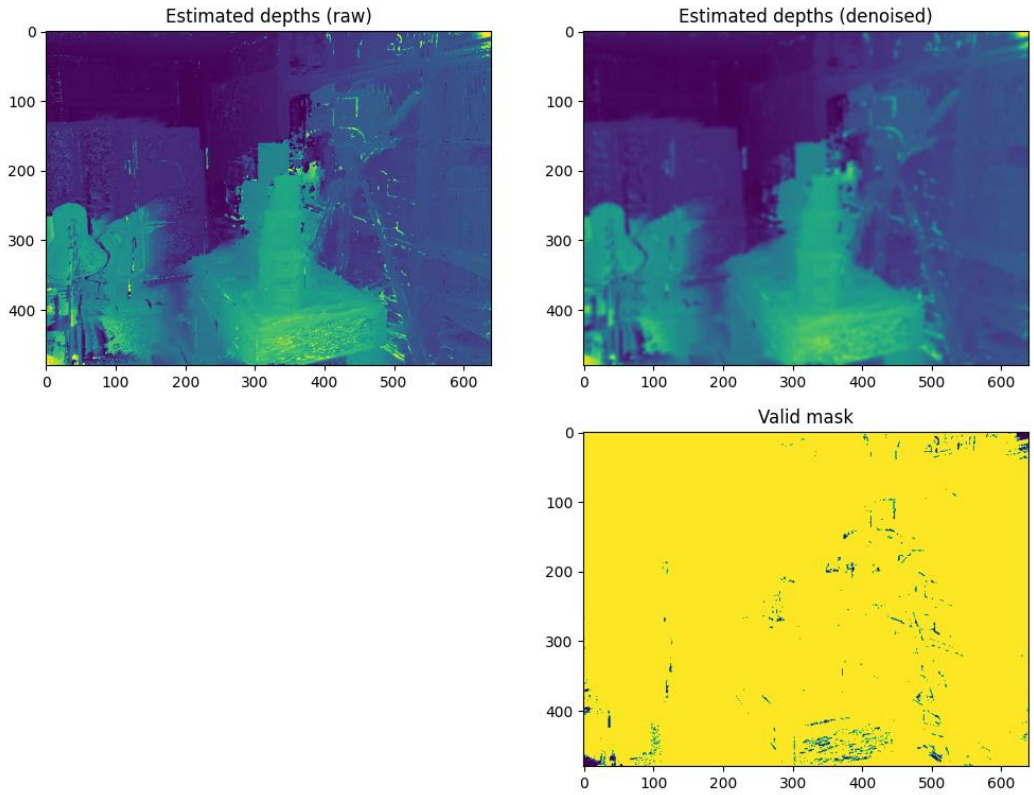

Figure 1. Post Processed Depths vs Raw Depths

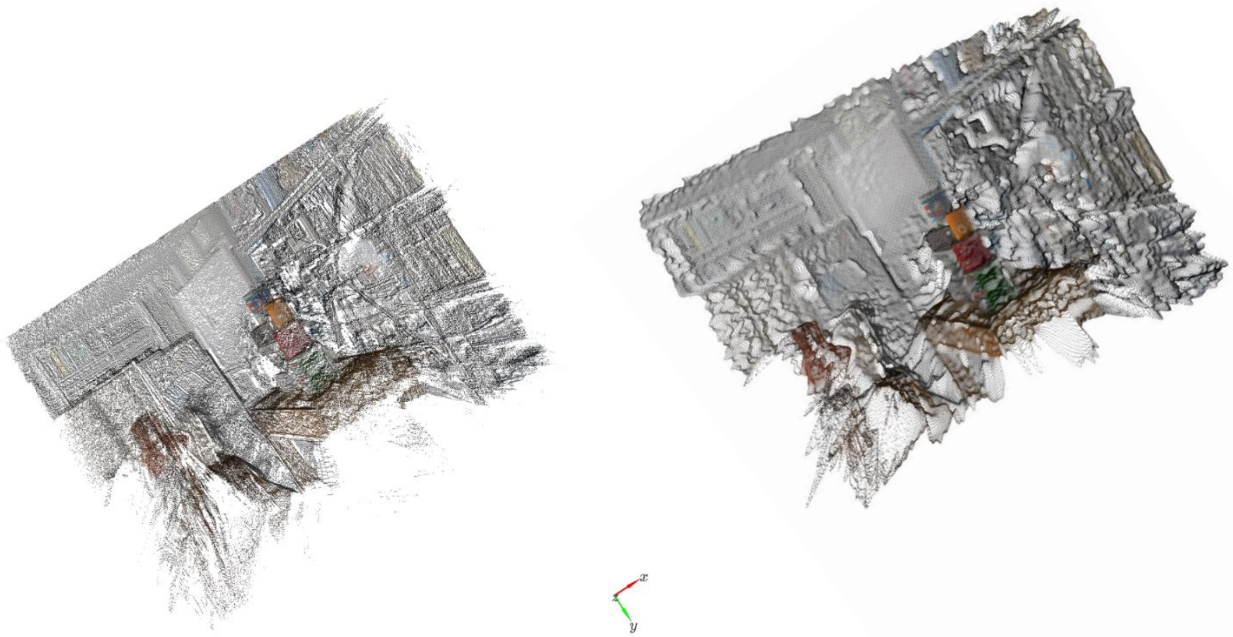

Figure 2. Raw point cloud 3D model

Figure 3. Post Processed Point Cloud 3D Model