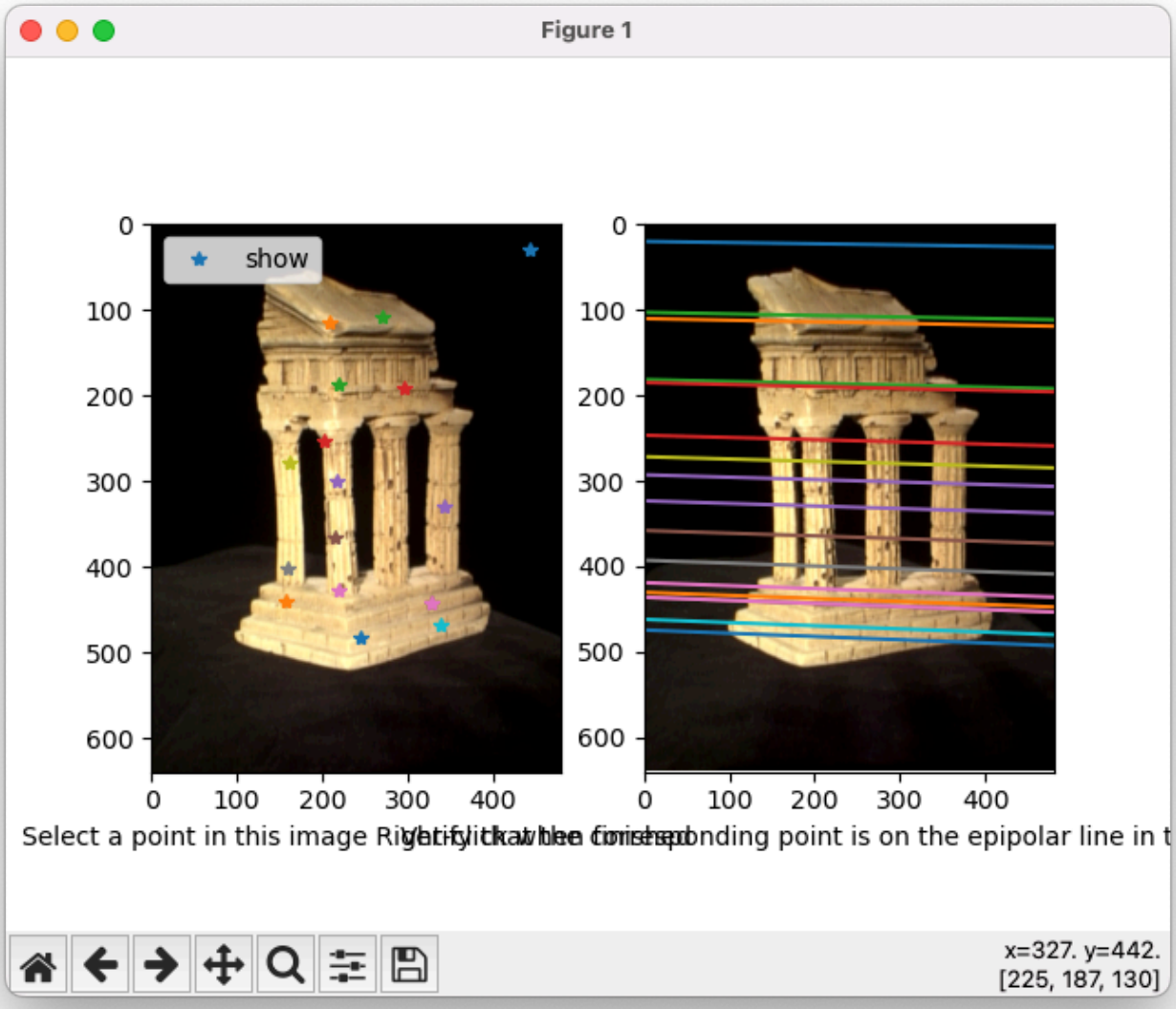
Project 5

Rongsheng Qian 301449387

3.1 Sparse reconstruction

3.1.1 Implement the eight point algorithm (2 pts)

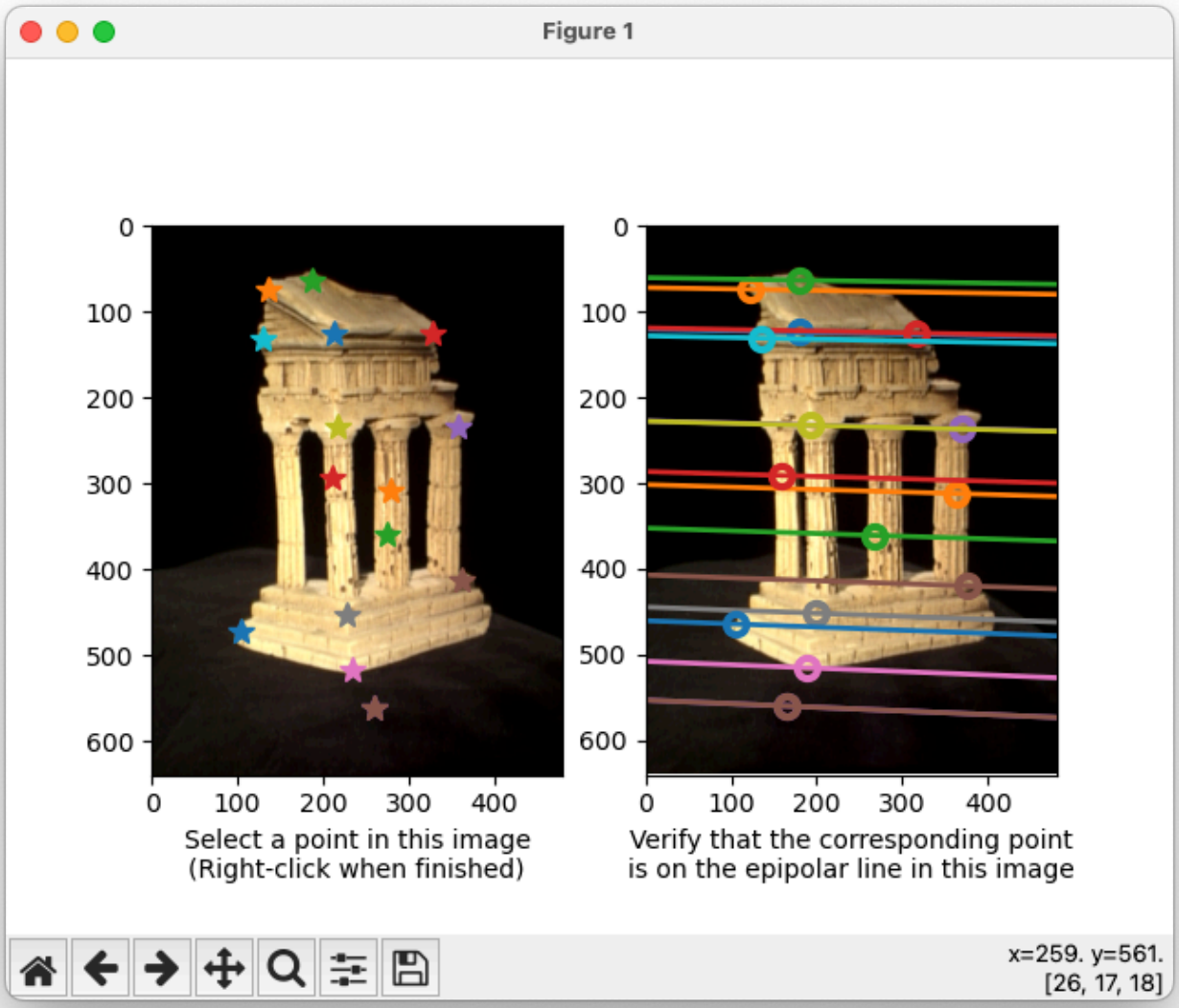
```
F =[[ 3.37554374e-09 -5.77488650e-08 -1.44640637e-05]
 [-1.29390178e-07 -1.48285427e-09  1.12458135e-03]
 [ 3.25298454e-05 -1.08034741e-03 -4.55969209e-03]]
```



3.1.2 Find epipolar correspondences (2 pts)

The similarity metric you decided to use is NCC. It is robust to Lighting and Shading Variations and can find instances of a smaller template within a larger image.

However it is sensitive to noise in the data. In the following instance, the orange/red point on the pillar in image 1 is matched on the other pilllar in image 2. The details of the pillars are similar. In this case, the matching results are easily confused or affected by noise.



3.1.3 Write a function to compute the essential matrix (2 pts)

```
E = [[ 0.00780296 -0.13397612 -0.04211513]
 [-0.30018241 -0.00345263  1.6557511 ]
 [ 0.00244448 -1.67570084 -0.00192676]]
```

3.1.4 Implement triangulation (2 pts)

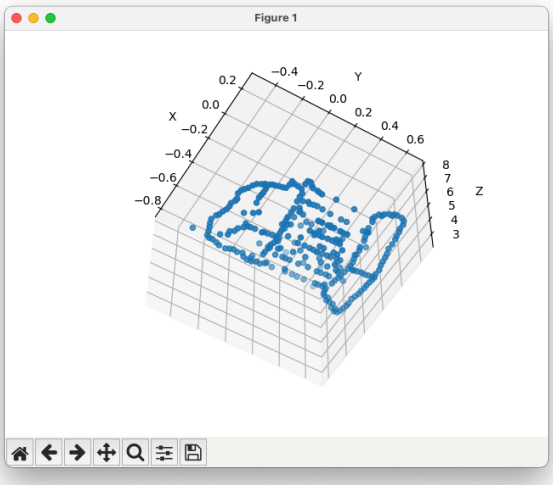
I determined which extrinsic matrices are correct by counting the 3D points which are in front of both cameras (positive depth) in each condition. I picked the one for which most of the 3D points are in front of both cameras (positive depth).

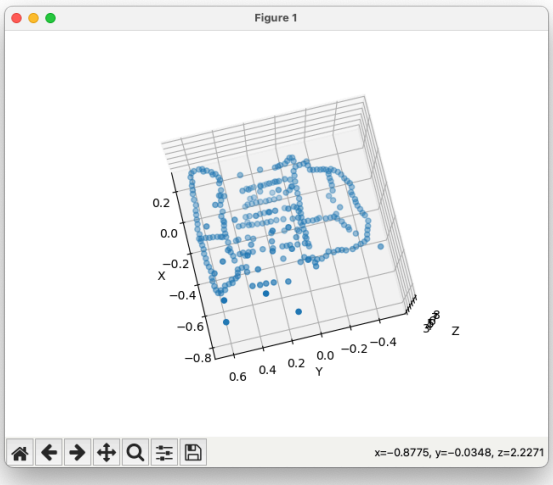
```
for i in range(P2_possible_e.shape[-1]):
    temp_p2_e = P2_possible_e[:,:,i]
    temp_P2 = np.dot(k2,temp_p2_e)
    points = triangulate(P1, pts1, temp_P2, pts2)
    front_count = 0
    for point in points:
        if point[-1]>0:
            front_count += 1
    if front_count > best:
        best = front_count
        P2 = temp_P2
        R2 = temp_p2_e[:,:-1]
        t2 = temp_p2_e[:,-1].reshape(-1,1)
        result_points = points
```

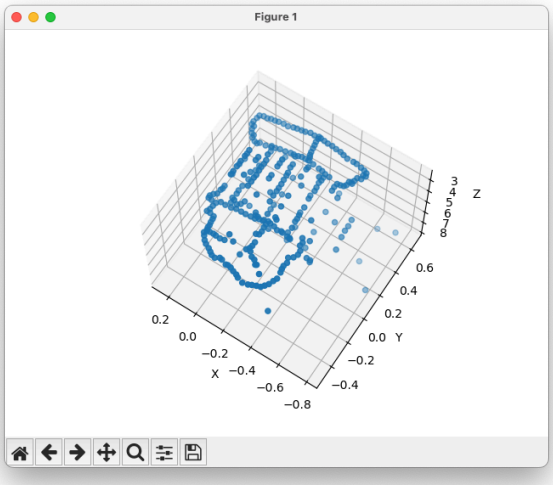
re-projection error

```
img1 re-projection error: 0.5527583965853167
img2 re-projection error: 0.5536822966377606
mean re-projection error: 0.5532203466115386
```

3.1.5 Write a test script that uses templeCoords (2 pts)

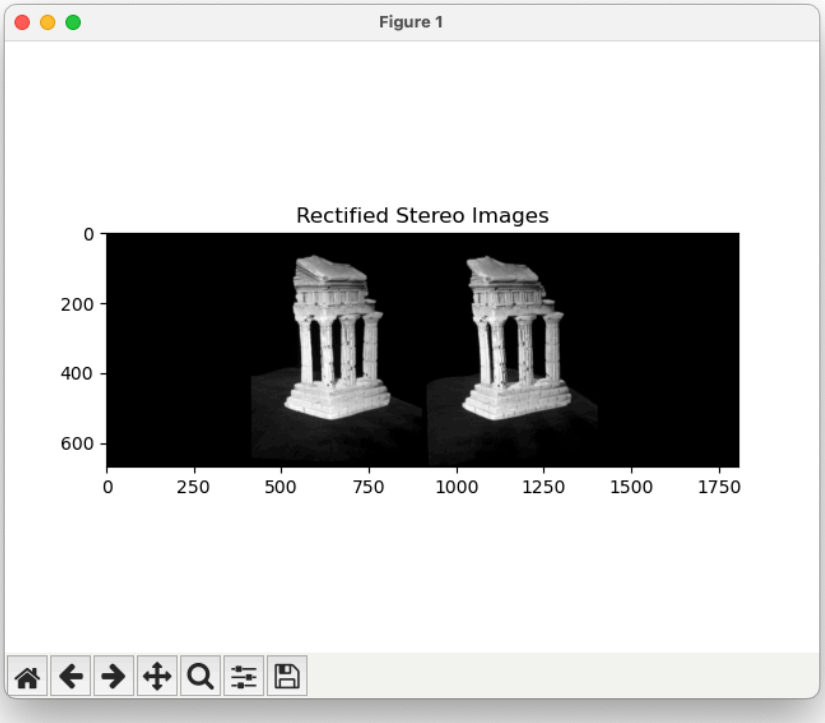


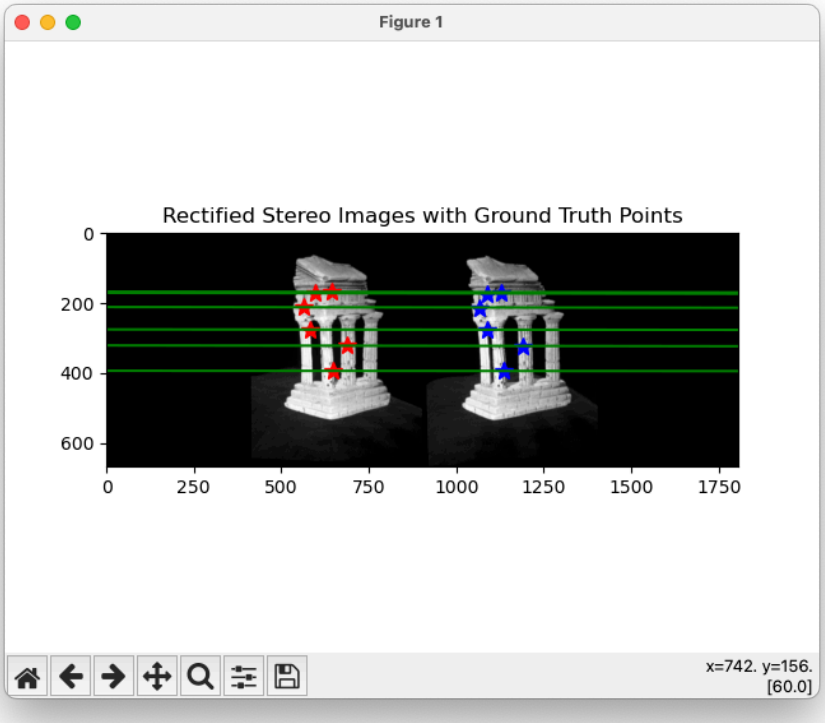




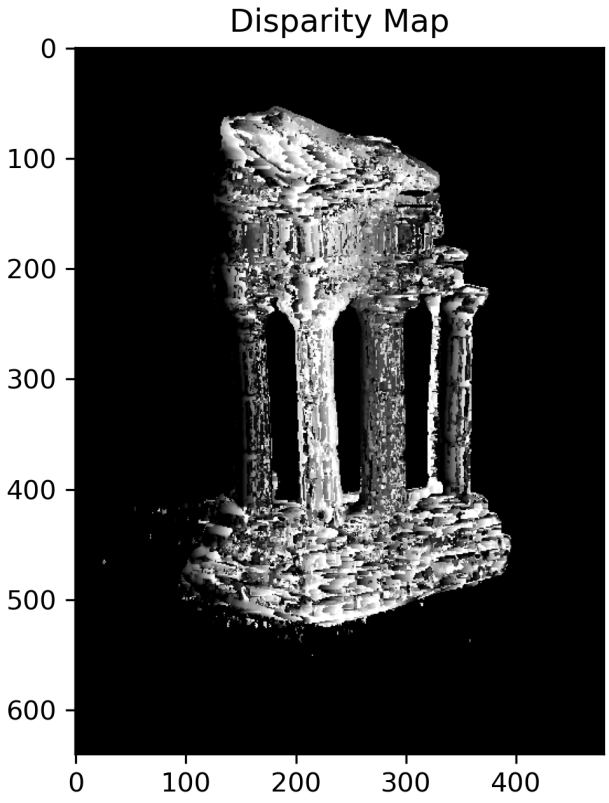
3.2 Dense reconstruction

3.2.1 Image rectification (2 pts)

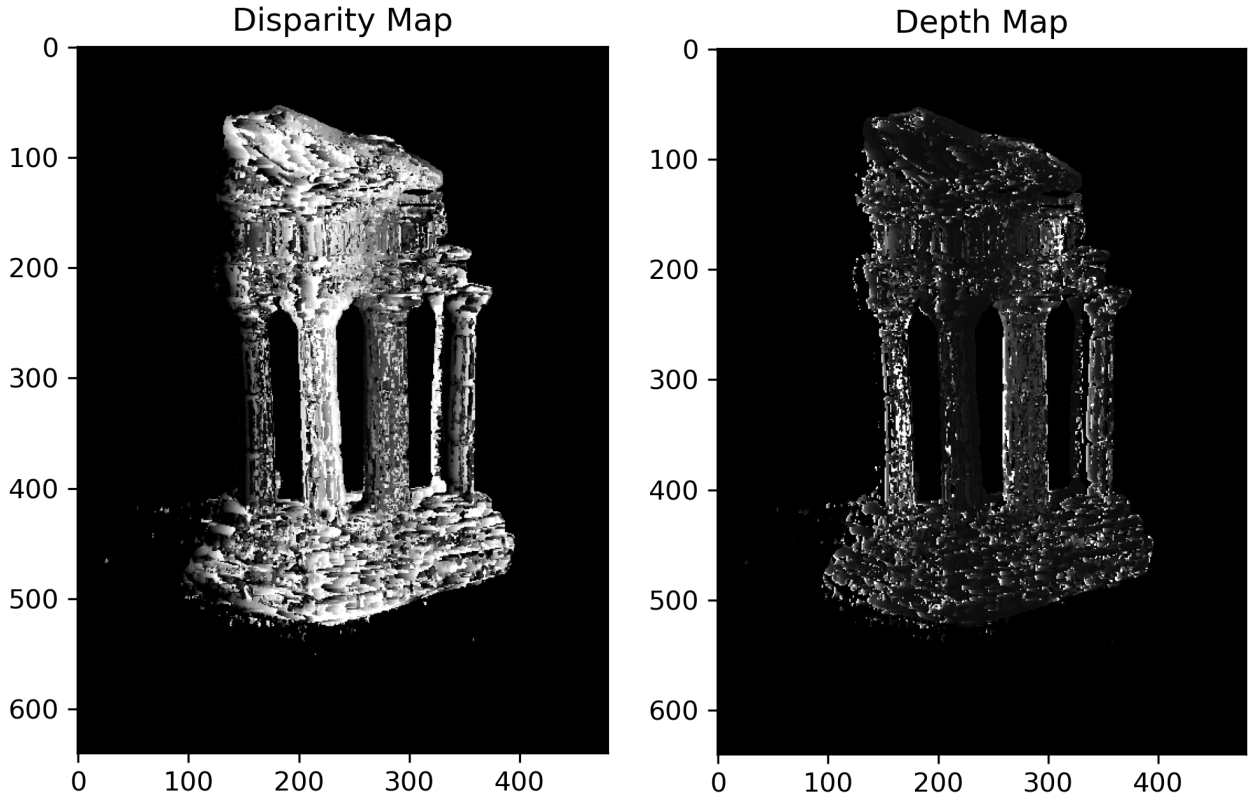




3.2.2 Dense window matching to find per pixel density (2 pts)



3.2.3 Depth map (2 pts)



3.3 Pose estimation

3.3.1 Estimate camera matrix P (2 pts)

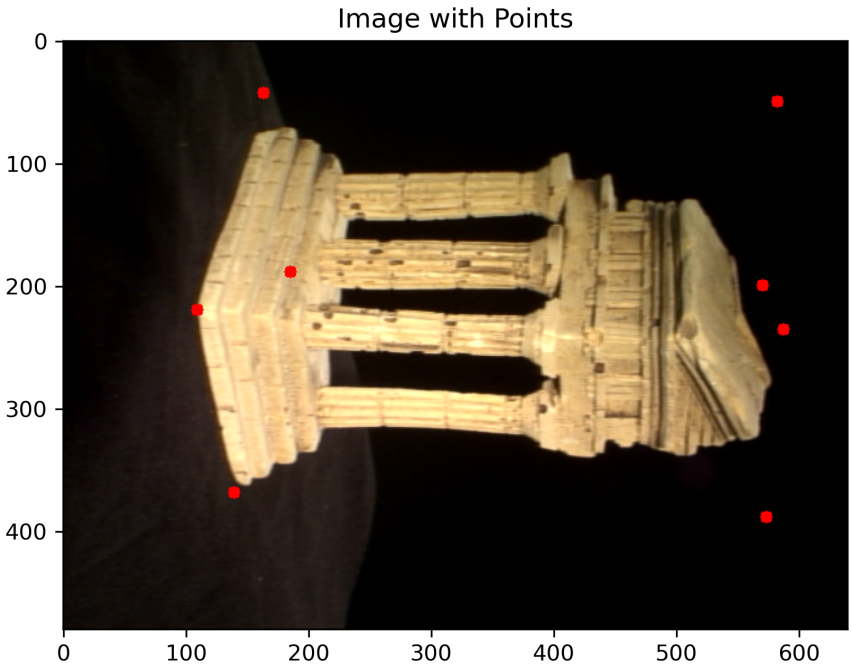
```
Reprojected Error with clean 2D points is 0.0000
Pose Error with clean 2D points is 0.0000
------------------------------
Reprojected Error with noisy 2D points is 3.1531
Pose Error with noisy 2D points is 0.0135

Reprojected Error with clean 2D points is 0.0000
Pose Error with clean 2D points is 0.0000
------------------------------
Reprojected Error with noisy 2D points is 2.2296
Pose Error with noisy 2D points is 0.0208

Reprojected Error with clean 2D points is 0.0000
Pose Error with clean 2D points is 0.0000
------------------------------
Reprojected Error with noisy 2D points is 3.2137
Pose Error with noisy 2D points is 2.0240
```

3.3.2 Estimate intrinsic/extrinsic parameters (1 pts)

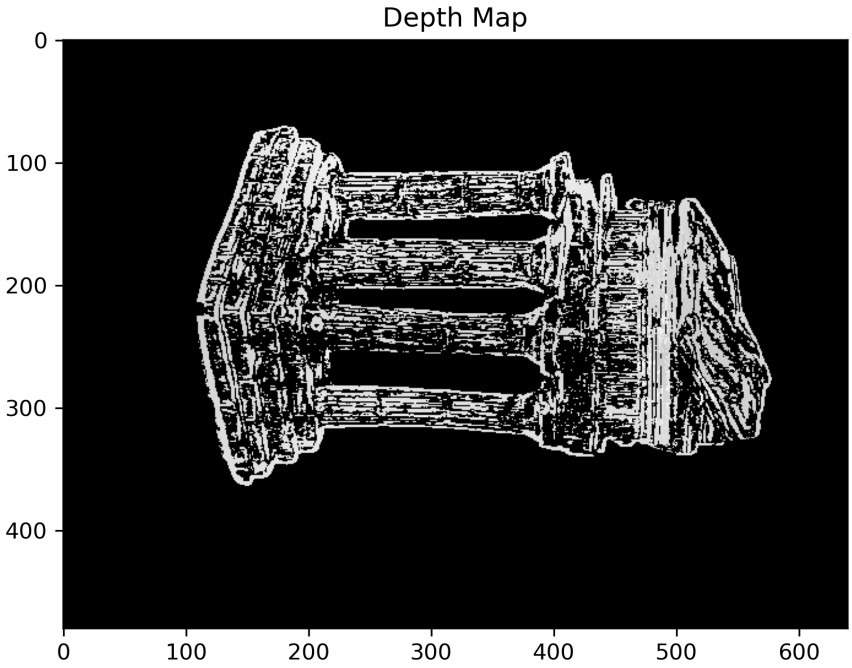
```
Intrinsic Error with clean 2D points is 0.0000
Rotation Error with clean 2D points is 0.0000
Translation Error with clean 2D points is 0.0000
------------------------------
Intrinsic Error with noisy 2D points is 0.6822
Rotation Error with noisy 2D points is 0.0755
Translation Error with noisy 2D points is 0.1660

Intrinsic Error with clean 2D points is 0.0000
Rotation Error with clean 2D points is 0.0000
Translation Error with clean 2D points is 0.0000
------------------------------
Intrinsic Error with noisy 2D points is 0.8108
Rotation Error with noisy 2D points is 0.0349
Translation Error with noisy 2D points is 0.3636

Intrinsic Error with clean 2D points is 0.0000
Rotation Error with clean 2D points is 0.0000
Translation Error with clean 2D points is 0.0000
------------------------------
Intrinsic Error with noisy 2D points is 0.6813
Rotation Error with noisy 2D points is 0.0531
Translation Error with noisy 2D points is 0.1008
```

3.4 Multi-view stereo

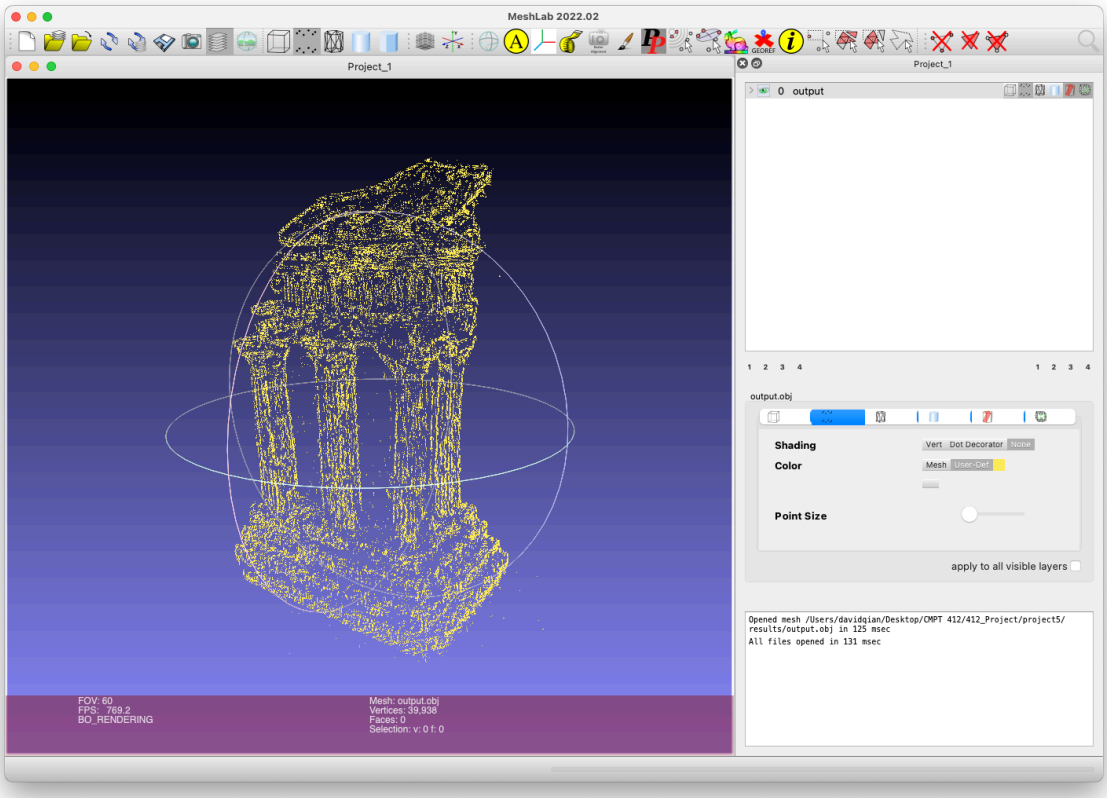
3.4.1 (1 pts)

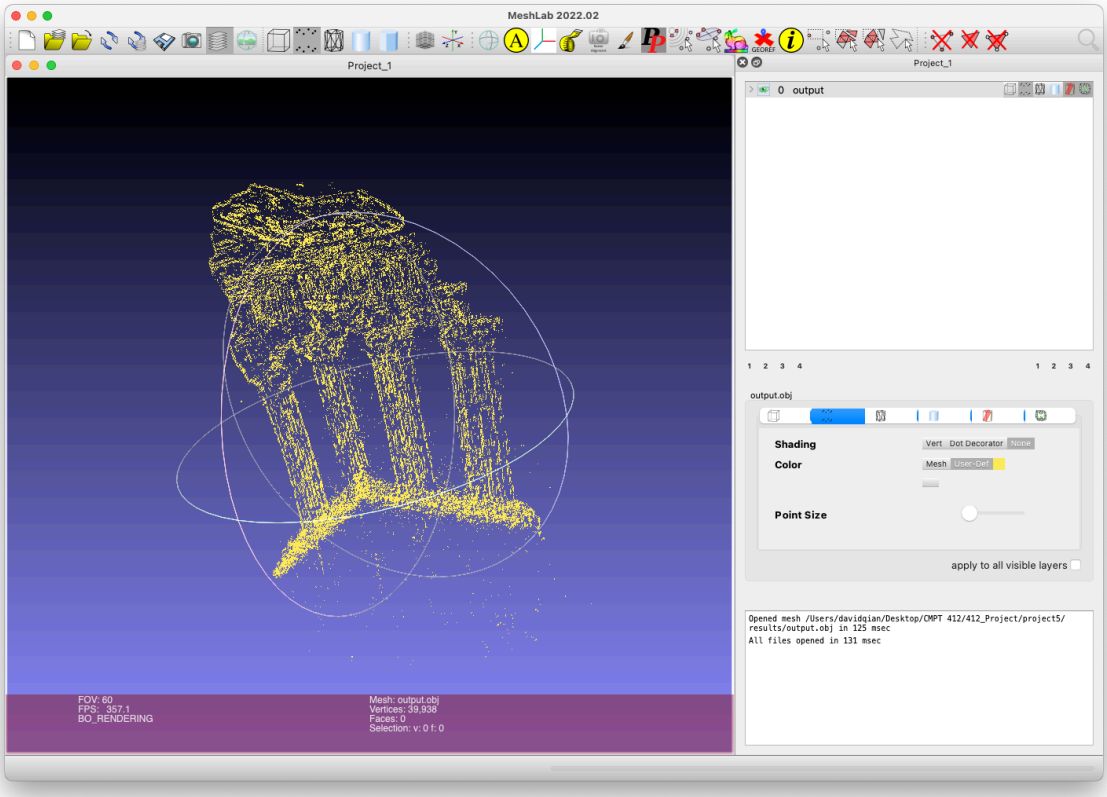


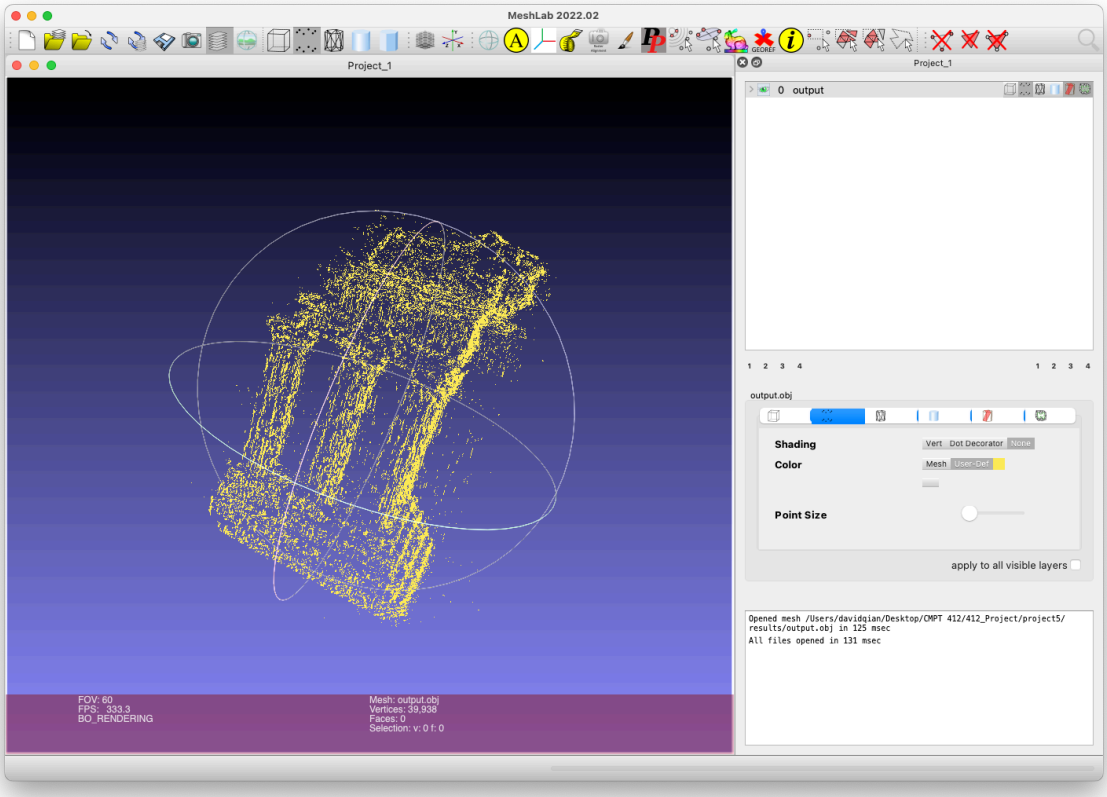
3.4.2 (1 pts) Here is my image[0] depthmap

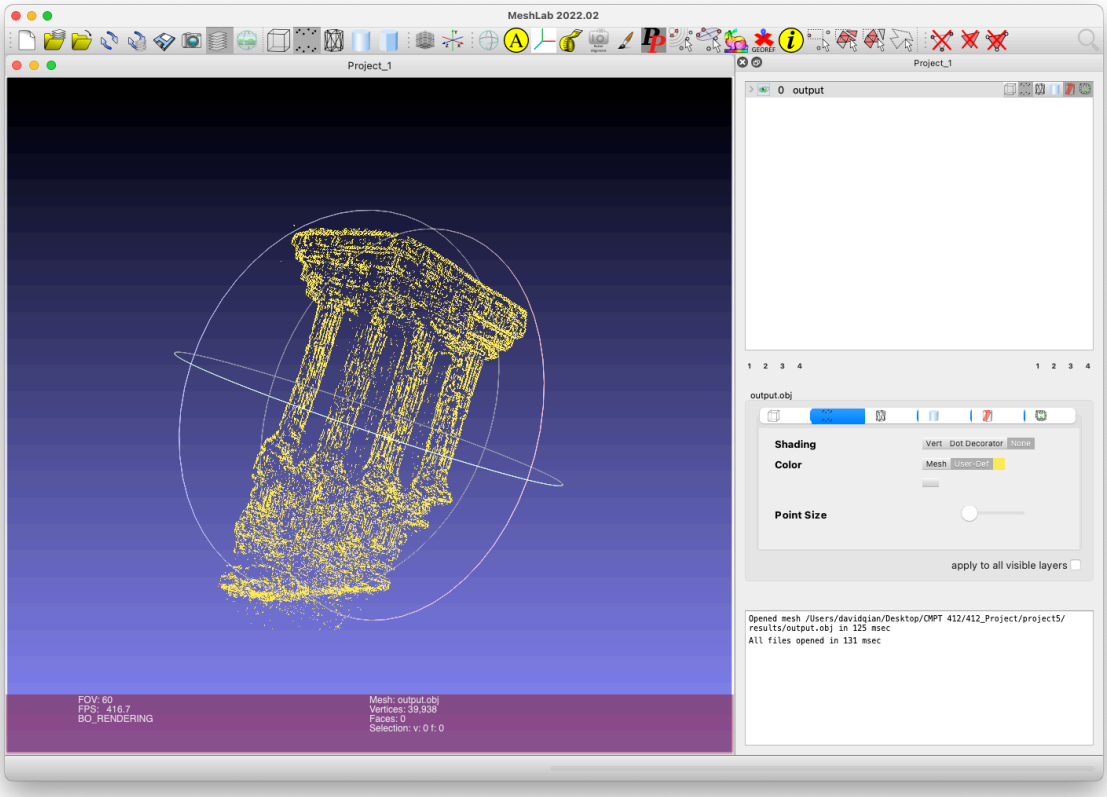


3.4.3 (1 pts)


MeshLab 2022.02
Project_1
Project_1
0 output
output.obj
Shading
Color
Point Size
apply to all visible layers
Opened mesh /Users/davidqian/Desktop/CMPT 412/412_Project/project5/results/output.obj in 125 msec
All files opened in 131 msec
FOV: 60
FPS: 769.2
BO_RENDERING
Mesh: output.obj
Vertices: 39,938
Faces: 0
Selection: v: 0 f: 0


MeshLab 2022.02
Project_1
Project_1
0 output
output.obj
Shading
Color
Point Size
apply to all visible layers
Opened mesh /Users/davidqian/Desktop/CMPT 412/412_Project/project5/results/output.obj in 125 msec
All files opened in 131 msec
FOV: 60
FPS: 357.1
BO_RENDERING
Mesh: output.obj
Vertices: 39,938
Faces: 0
Selection: v: 0 f: 0


MeshLab 2022.02
Project_1
Project_1
0 output
output.obj
Shading
Color
Point Size
apply to all visible layers
Opened mesh /Users/davidqian/Desktop/CMPT 412/412_Project/project5/results/output.obj in 125 msec
All files opened in 131 msec
FOV: 60
FPS: 333.3
BO_RENDERING
Mesh: output.obj
Vertices: 39,938
Faces: 0
Selection: v: 0 f: 0


MeshLab 2022.02
Project_1
Project_1
0 output
output.obj
Shading
Color
Point Size
apply to all visible layers
Opened mesh /Users/davidqian/Desktop/CMPT 412/412_Project/project5/results/output.obj in 125 msec
All files opened in 131 msec
FOV: 60
FPS: 416.7
BO_RENDERING
Mesh: output.obj
Vertices: 39,938
Faces: 0
Selection: v: 0 f: 0