CS682 Computer Vision

P4 Assignment Write Up

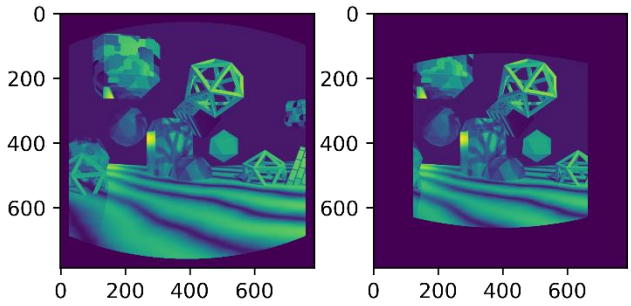
Mithilaesh Jayakumar

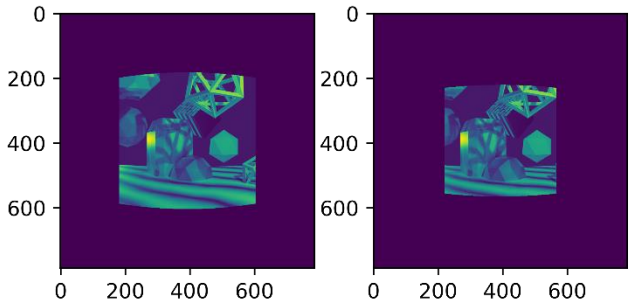
G01206238

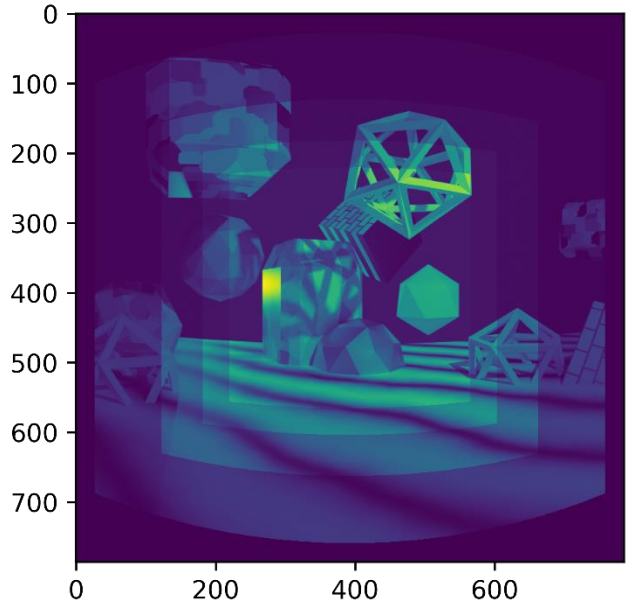
P4.1 Spherical Reprojection

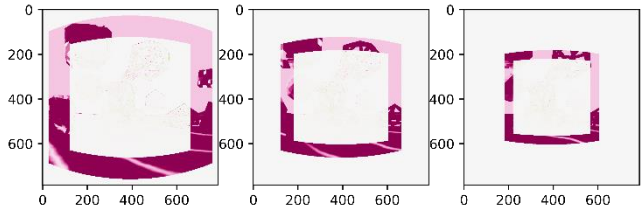
The below are the output images and plots generated for the spherical reprojection of the four images generated with different focal lengths using blender.

```
2 * Field of View: 1.5707963267948966
2 * Field of View: 1.5707963267948966
2 * Field of View: 1.5707963267948966
2 * Field of View: 1.5707963267948966
```



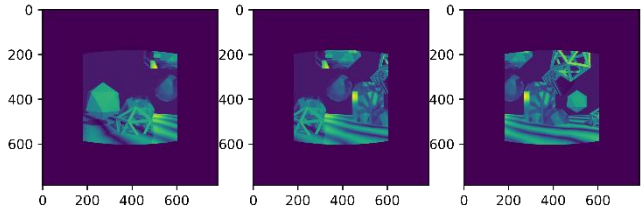


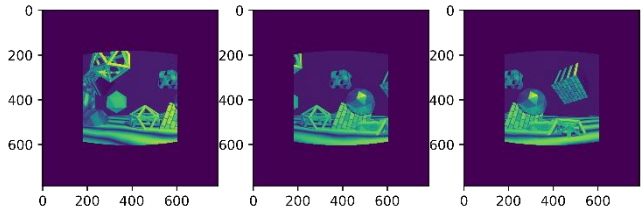




P4.2 Panorama Stitching

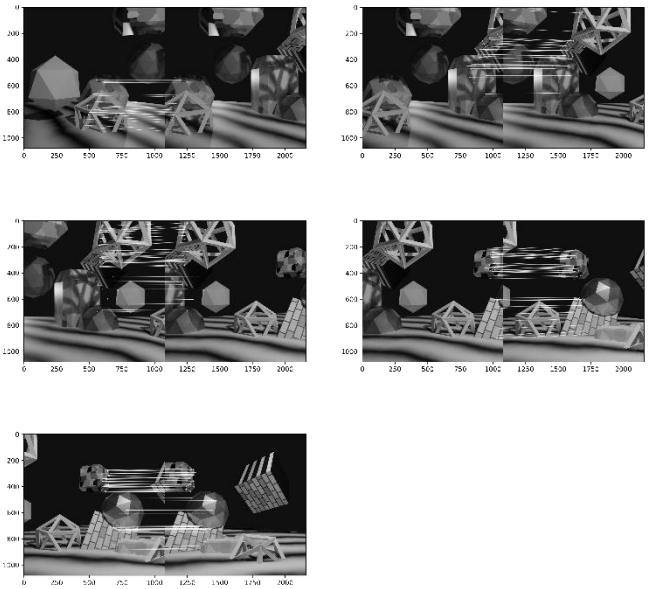
The shows the spherical reprojection of the six images that have the camera rotated along z axis from 40 to -60 degrees with an offset of -20 degrees.



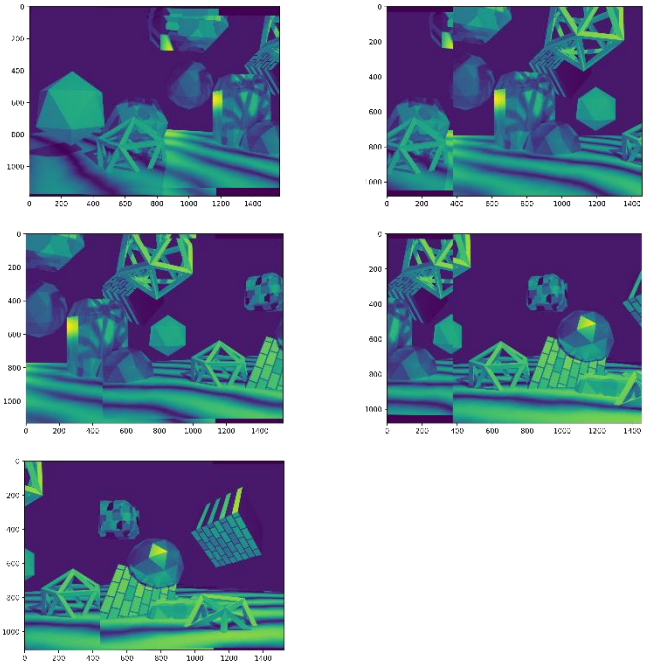


The below are the plots for step by step generation of the panorama of all six images.

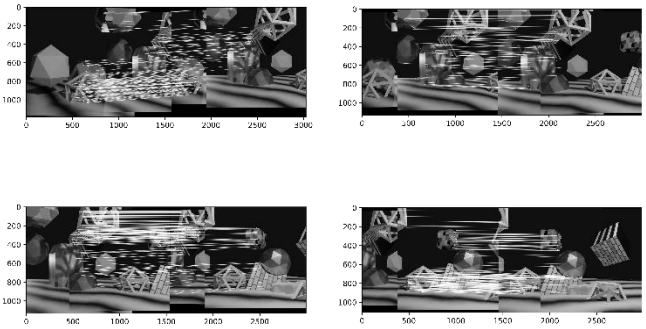
Matching 2 Consecutive Images



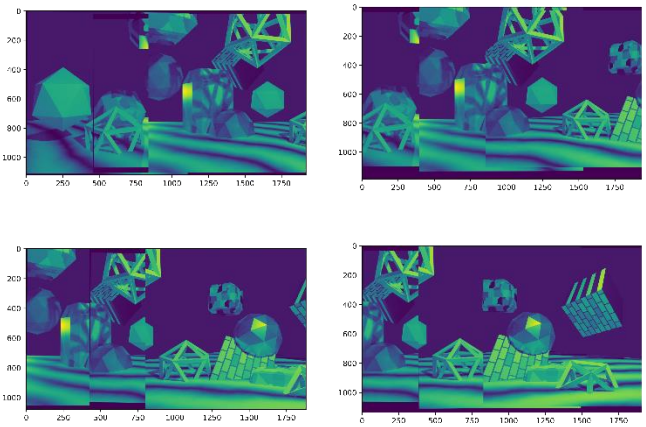
Joining 2 Consecutive Images



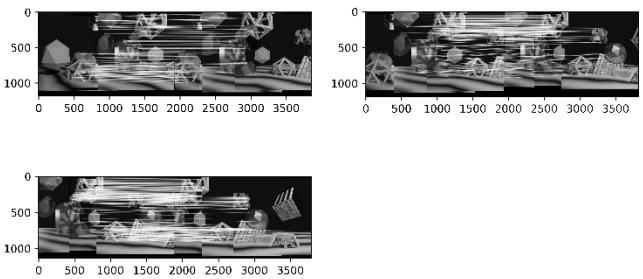
Matching 3 Consecutive Images



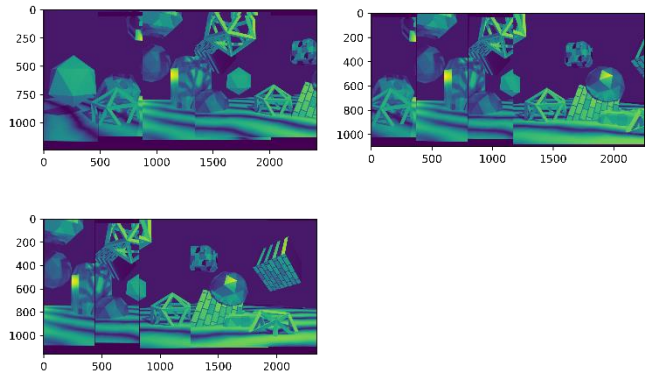
Joining 3 Consecutive Images



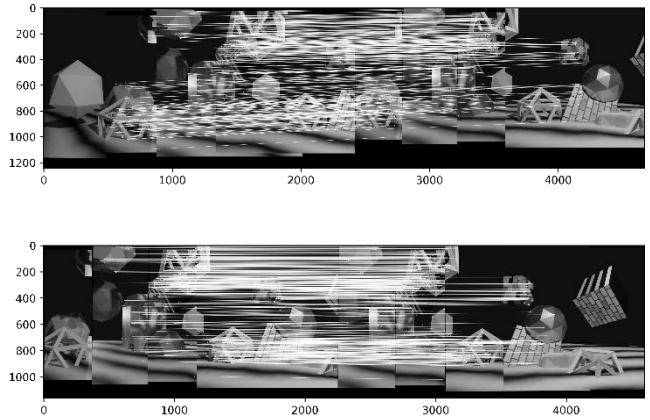
Matching 4 Consecutive Images



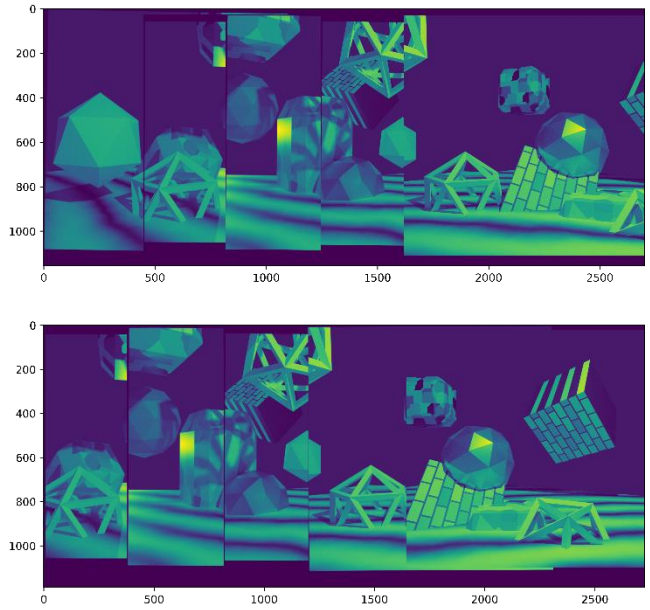
Joining 4 Consecutive Images



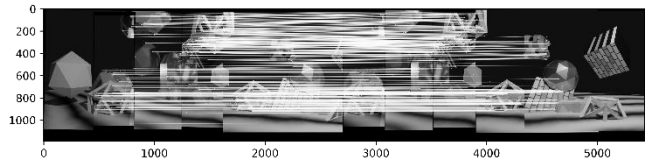
Matching 5 Consecutive Images



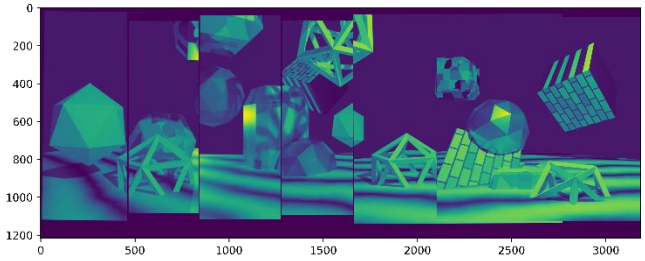
Joining 5 Consecutive Images



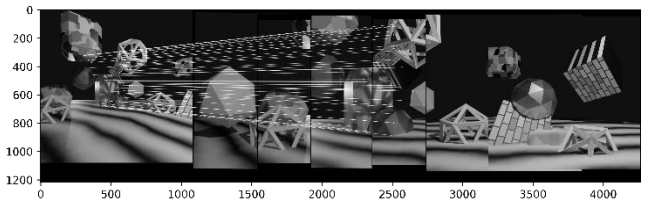
Matching All 6 Images

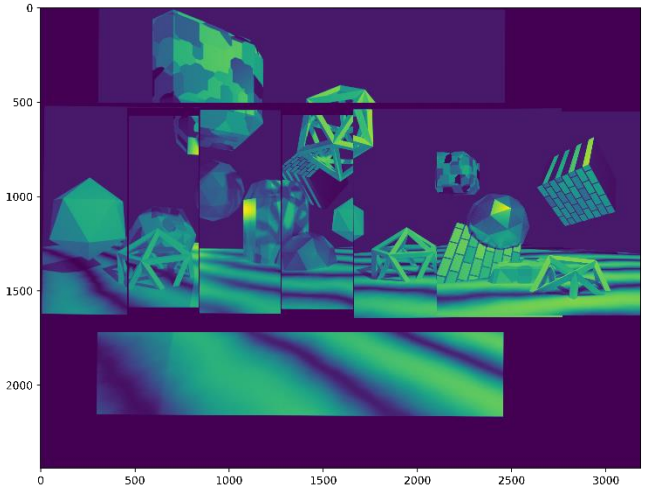


Joining All 6 Images



The below is the plot of combining the image of focal length 20mm with the panorama image.





P4.3 Triangulation

P4.3.1 Projecting Into Image Space

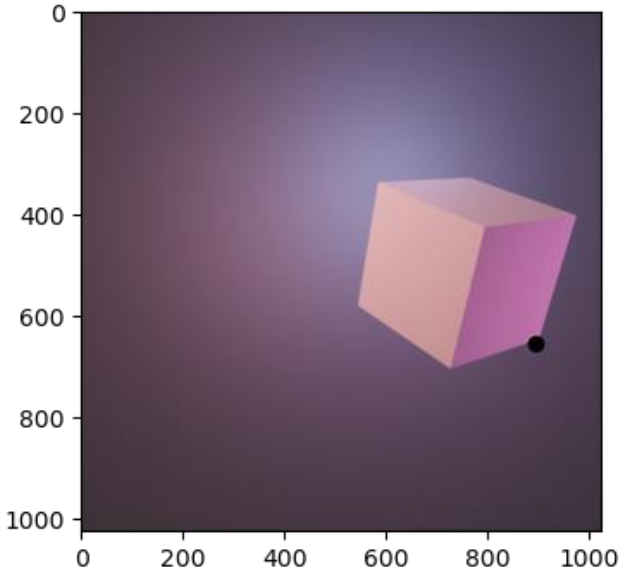
The camera matrix computation for the two images is defined as shown below.

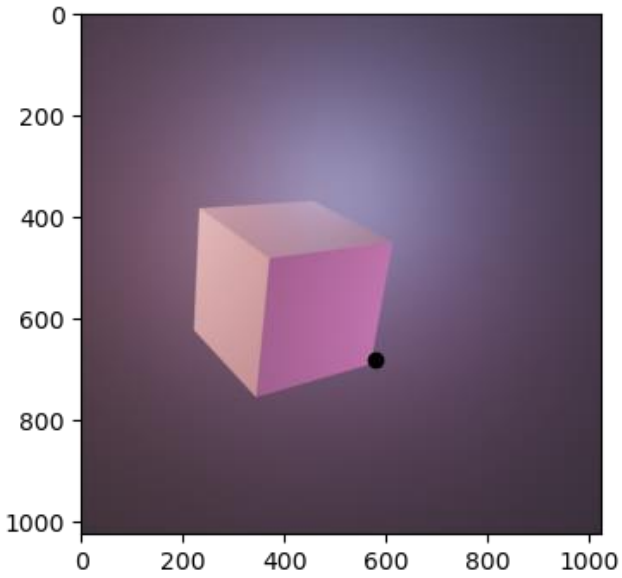
```
# TASK: Implement the camera matrices & get_projected_point
f = 1137.8
a = image_a[514][514]
b = image_b[514][514]
intrinA = np.array([[(f*a[0])/a[2], 0, a[0]], [0, (f*a[1])/a[2], a[1]], [0, 0, 1]])
intrinB = np.array([[(f*b[0])/b[2], 0, b[0]], [0, (f*b[1])/b[2], b[1]], [0, 0, 1]])
extrinA = np.array([[1, 0, 0, 0], [0, 1, 0, -0.2], [0, 0, 1, 5]])
extrinB = np.array([[1, 0, 0, -1.5], [0, 1, 0, 0], [0, 0, 1, 5]])

Pa =  np.dot(intrinA,extrinA)
Pb = np.dot(intrinB,extrinB)
```

The below images show the projection of the 3D point into the image space. The given 3D point overlaps with the bottom right most corner of the cube in both the images.

```
[892.54948942 651.47618952   4.48180594]
[576.96686505 679.73466111   4.48180594]
```





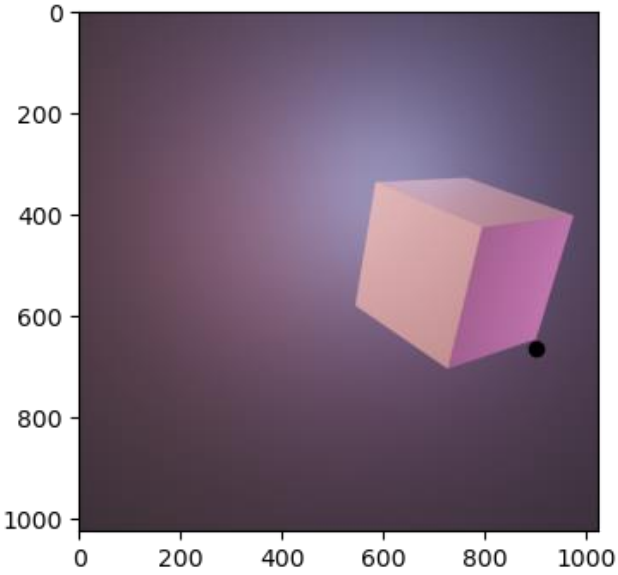
P4.3.2 Determining the Size of the Cube

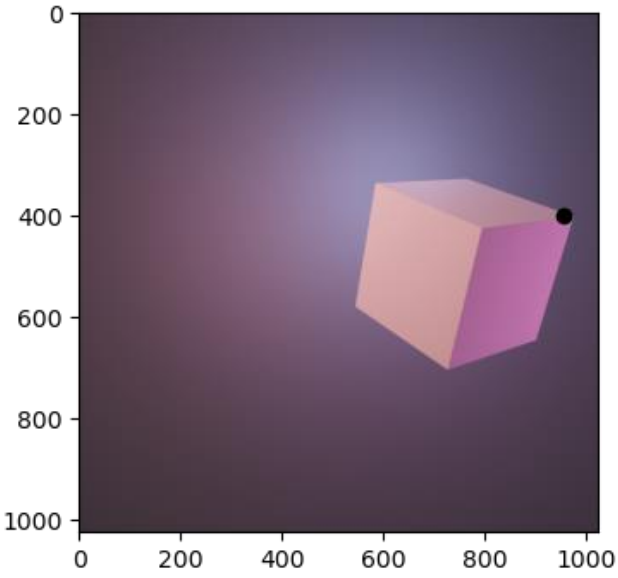
The following are the 2D and corresponding 3D coordinates for the specified corner of the cube.

```
[901.71790874 664.02109349   4.48180594]
[954.4636041  399.53910674   4.48180594]
```

X0 = np.array([0.85244616, 0.9508618, -0.51819406, 1])

X1 = np.array([0.90244616, 0.6508618, -0.51819406, 1]

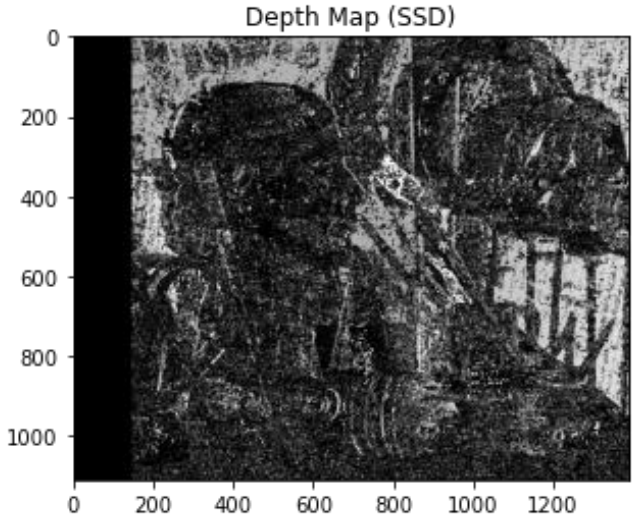


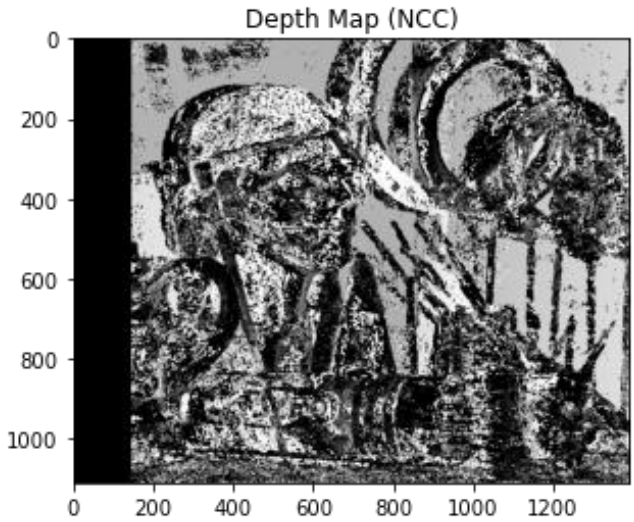


The side of the cube is approximately around 290 to 320 mm.

P4.4 Stereo Patch Matching

The depth plots generated by the two methods are shown below.





The bottom left corner of both the depth plots are noisy and not accurate because of the presence of shadows in the image. This case is because of specularities and this can be improved by removal of shadows.