Name: Janmejay Mohanty

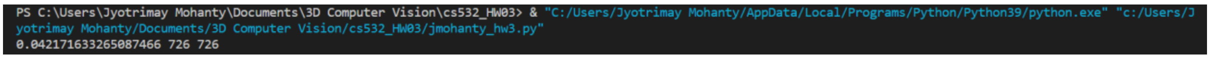
CS 532-A

HW-3 Report

Source Code:

```
import numpy as np

from PIL import Image

import open3d as o3d


# Assigning and creating an array list.


P = np.array([[776.649963,-298.408539,-32.048386,993.1581875,132.852554,120.885834,-
759.210876,1982.174000,0.744869,0.662592,-0.078377,4.629312012],

   [431.503540,586.251892,-137.094040,1982.053375,23.799522,1.964373,-
657.832764,1725.253500,-0.321776,0.869462,-0.374826,5.538025391],

   [-153.607925,722.067139,-127.204468,2182.4950,141.564346,74.195686,-
637.070984,1551.185125,-0.769772,0.354474,-0.530847,4.737782227],

   [-823.909119,55.557896,-82.577644,2498.20825,-31.429972,42.725830,-
777.534546,2083.363250,-0.484634,-0.807611,-0.335998,4.934550781],

   [-715.434998,-351.073730,-147.460815,1978.534875,29.429260,-2.156084,-
779.121704,2028.892750,0.030776,-0.941587,-0.335361,4.141203125],

   [-417.221649,-700.318726,-27.361042,1599.565000,111.925537,-169.101776,-
752.020142,1982.983750,0.542421,-0.837170,-0.070180,3.929336426],

   [94.934860,-668.213623,-331.895508,769.8633125,-549.403137,-58.174614,-
342.555359,1286.971000,0.196630,-0.136065,-0.970991,3.574729736],

   [452.159027,-658.943909,-279.703522,883.495000,-262.442566,1.231108,-
751.532349,1884.149625,0.776201,0.215114,-0.592653,4.235517090]])


# Declaring and assigning the values to variables, list, etc.

dict_c = {}

dict_s = {}

matrix_p = np.zeros((3,4,8))
```

```
# Fetching the required image data files.
for i in range(8):
    dict_c[i] = np.asarray(Image.open("cam0" + str(i) + "_00023_0000008550.png"))
    dict_s[i] = np.asarray(Image.open("silh_cam0" + str(i) + "_00023_0000008550.pbm"))
    matrix_p[:,:,i] = np.reshape(P[i],(3,4))
range_x = 5
range_y = 6
range_z = 2.5
volume = range_x*range_y*range_z
v_num = 1000000                    # Vox Number
v_size = np.power((volume/v_num),1/3)   # Vox Size
v_grid = []
s_grid = []                        # Surf Grid
d_grid = []
for x in np.arange(-2.5, 2.5, v_size):
    for y in np.arange(-3, 3, v_size):
        min_z = 3
        max_z = -3
        for z in np.arange(0, 2.5, v_size):
            pass_mat = np.zeros(8)
            co_ordinate = [x, y, z, 1]
            for i in range(8):
                pt = np.dot(co_ordinate,np.transpose(matrix_p[:,:,i]))
                pt = pt/pt[2]
                if((0<=pt[1]<582) and (0<=pt[0]<780)):
                    pass_mat[i] = dict_s[i][int(pt[1]),int(pt[0])]
            if(np.sum(pass_mat) == 8):
                v_grid.append([x,y,z])
                if(z<min_z):
                    min_z = z
                if(z>max_z):
                    max_z = z
        if(min_z!=3 and max_z!=-3):
            s_grid.append([x,y,min_z])
            s_grid.append([x,y,max_z])
            d_grid.append('zbot')
            d_grid.append('ztop')
# Declaring a function for Grid Cleared.
def grid_cleared(vox,surf):
    for s in surf:
        if(s in vox):
            vox.remove(s)
grid_cleared(v_grid,s_grid)
for z in np.arange(0, 2.5, v_size):
    for y in np.arange(-3, 3, v_size):
        min_x = 3
        max_x = -3
        for x in np.arange(-2.5, 2.5, v_size):
            if([x,y,z] in v_grid):
                if(x<min_x):
                    min_x = x
                if(x>max_x):
                    max_x = x
        if(min_x!=3 and max_x!=-3):
            s_grid.append([min_x,y,z])
            s_grid.append([max_x,y,z])
            d_grid.append('xleft')
            d_grid.append('xright')
grid_cleared(v_grid,s_grid)
for z in np.arange(0, 2.5, v_size):
    for x in np.arange(-2.5, 2.5, v_size):
        min_y = 3
        max_y = -3
        for y in np.arange(-3, 3, v_size):
            if([x,y,z] in v_grid):
                if(y<min_y):
                    min_y = y
                if(y>max_y):
                    max_y = y
        if(min_z!=3 and max_z!=-3):
            s_grid.append([x,min_y,z])
            s_grid.append([x,max_y,z])
            d_grid.append('yback')
            d_grid.append('yfront')
grid_color = []
dict_d = {
    "ztop": 6,
    "zbot": 3,
    "yback": 3,
    "yfront": 0,
    "xright": 2,
    "xleft": 5
}
for i in range(0,len(d_grid)):
    view = dict_d[d_grid[i]]
    co_ordinate = s_grid[i]+[1]
    pt = np.dot(co_ordinate,np.transpose(matrix_p[:,:,view]))
    pt = pt/pt[2]
    rgb = dict_c[view][int(pt[1]),int(pt[0])]
    grid_color.append([float(rgb[0])/255,float(rgb[1])/255,float(rgb[2])/255])
print(v_size,len(grid_color),len(s_grid))
pcd = o3d.geometry.PointCloud()
pcd.points = o3d.utility.Vector3dVector(s_grid)
pcd.colors = o3d.utility.Vector3dVector(grid_color)

o3d.io.write_point_cloud("./dancer.ply", pcd)

o3d.visualization.draw_geometries([pcd])
```



```
PS C:\Users\Jyotrimay Mohanty\Documents\3D Computer Vision\cs532_HW03> & "C:/Users/Jyotrimay Mohanty/AppData/Local/Programs/Python/Python39/python.exe" "c:/Users/J
yotrimay Mohanty/Documents/3D Computer Vision/cs532_HW03/jmohanty_hw3.py"
0.042171633265087466 726 726
```