hw2_p2

February 16, 2018

In [1]:
```
import numpy as np
import pandas as pd
import tensorflow as tf

%matplotlib inline
import matplotlib
import matplotlib.pyplot as plt

import pymesh
from pyntcloud import PyntCloud
```

In [2]:
```
# compute triangle mesh surface area
def triangle_area(x):
    a = x[:,0,:] - x[:,1,:]
    b = x[:,0,:] - x[:,2,:]
    cross = np.cross(a, b)
    area = 0.5 * np.linalg.norm(np.cross(a, b), axis=1)
    return area

# compute euclidean distance matrix
def euclidean_distance_matrix(x):
    r = np.sum(x*x, 1)
    r = r.reshape(-1, 1)
    distance_mat = r - 2*np.dot(x, x.T) + r.T
    #return np.sqrt(distance_mat)
    return distance_mat

# update distance matrix and select the farthest point from set S after a new point is s
def update_farthest_distance(far_mat, dist_mat, s):
    for i in range(far_mat.shape[0]):
        far_mat[i] = dist_mat[i,s] if far_mat[i] > dist_mat[i,s] else far_mat[i]
    return far_mat, np.argmax(far_mat)

# initialize matrix to keep track of distance from set s
def init_farthest_distance(far_mat, dist_mat, s):
    for i in range(far_mat.shape[0]):
        far_mat[i] = dist_mat[i,s]
    return far_mat
```

In [3]:
```
# get sample from farthest point on every iteration
def farthest_point_sampling(obj_file, num_samples=1000):
    mesh = pymesh.load_mesh(obj_file)
    faces = mesh.vertices[mesh.faces]
    area = triangle_area(faces)
    total_area = np.sum(area)

    set_P = []
    for i in range(faces.shape[0]):
        num_gen = area[i] / total_area * 10000
        for j in range(int(num_gen)+1):
            r1, r2 = np.random.rand(2)
            d = (1-np.sqrt(r1)) * faces[i,0] + np.sqrt(r1)*(1-r2) * faces[i,1] + np.sqrt
            set_P.append(d)

    set_P = np.array(set_P)
    num_P = set_P.shape[0]

    distance_mat = euclidean_distance_matrix(set_P)

    set_S = []
    s = np.random.randint(num_P)
    far_mat = init_farthest_distance(np.zeros((num_P)), distance_mat, s)

    for i in range(num_samples):
        set_S.append(set_P[s])
        far_mat, s = update_farthest_distance(far_mat, distance_mat, s)

    return np.array(set_S)
```

In [4]:
```
teapot_pts = farthest_point_sampling('teapot.obj')
```

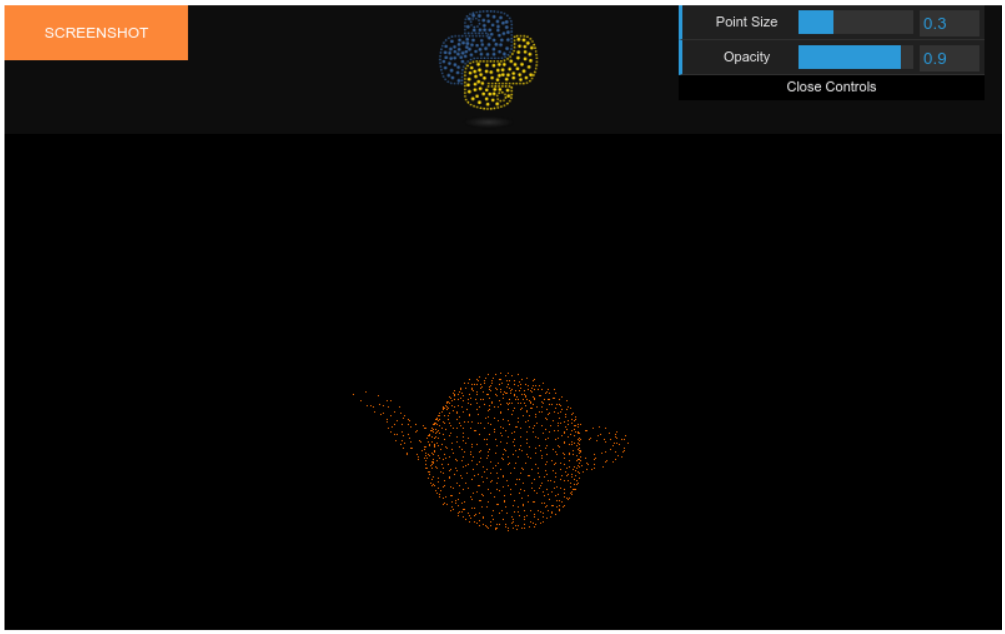
In [5]:
```
points = pd.DataFrame(teapot_pts, columns=['x', 'y', 'z'])
cloud = PyntCloud(points)
cloud.plot(line_color='')
```

Out[5]:
```
<IPython.lib.display.IFrame at 0x7fc5efb855f8>
```

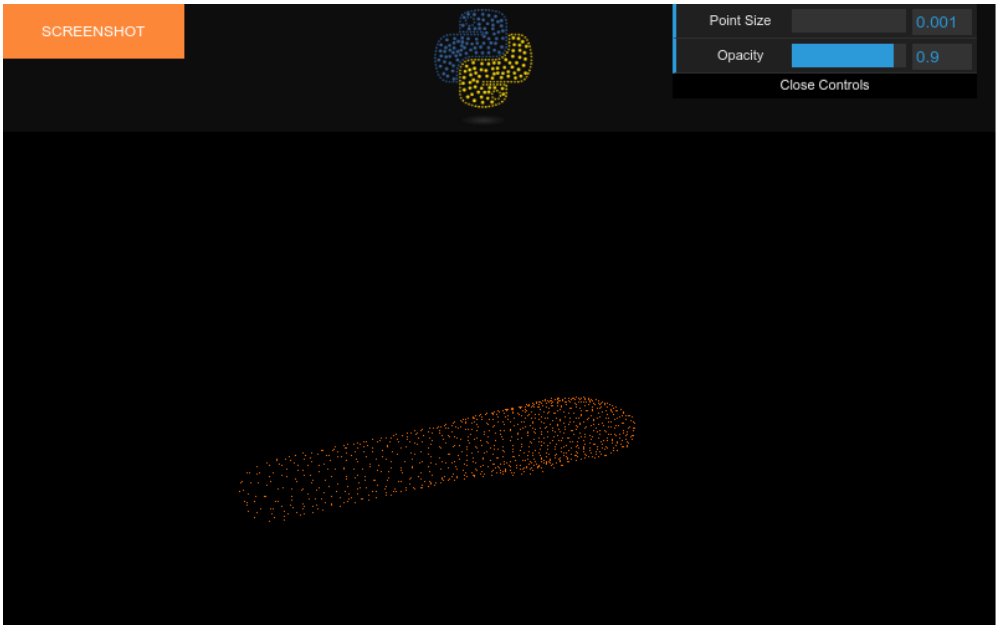
In [6]:
```
violin_pts = farthest_point_sampling('violin_case.obj')
```

In [7]:
```
points = pd.DataFrame(violin_pts, columns=['x', 'y', 'z'])
cloud = PyntCloud(points)
cloud.plot(line_color='')
```

Out[7]:
```
<IPython.lib.display.IFrame at 0x7fc5ef922630>
```



teapot



violin case