|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 课程编号 | 2706075 | | 课程名称 | | 三维视觉导论 | | 主讲教师 | | 方昊 | 评分 |  |
| 学 号 | 2110276001 | | 姓名 | 林杰锋 | | 专业年级 | | 21级电子信息1班 | | | |
|  | | | | | | | | | | | |
| 教师评语： | | | | | | | | | | | |
| 题目： | | **作业二** | | | | | | | |  | |

作业内容：

1. 实现基于region growing的点云数据平面检测算法
2. 改变region growing中的两个重要参数，观察平面检测结果的变化，并总结规律

作业要求：

1. 填充 （C++）
2. 要求完成实验报告，报告内容应该包括：
3. 背景知识介绍
4. 实验内容
5. 实现步骤

d. 结果分析 (可用MeshLab可视化实验结果，即ply文件，并截屏保存)

1. 提供源代码（作为附件）
2. 可参考课程Lecture\_3 slides 105，但不限于基本模型的实验，如有改进模型且效果良好有加分。
3. 页数不限。

**Report**

**1 Background**

About Primitive-based surface reconstruction/cloud segmentation, and how to extract geometric primitives from point cloud sets, there are several ways to have it done. Such as region growing, RANSAC, Accumulation methods, and Global regularities. But this time I am going to talk ablout region growing. Region growing is an iterative method of surface construction, which is also known as shape detection/point cloud segmentation. Here are some crucial steps of realization the algorithm of region growing and implying it to experiments.

Select a point and a primitive hypothesis(Fig. 1).

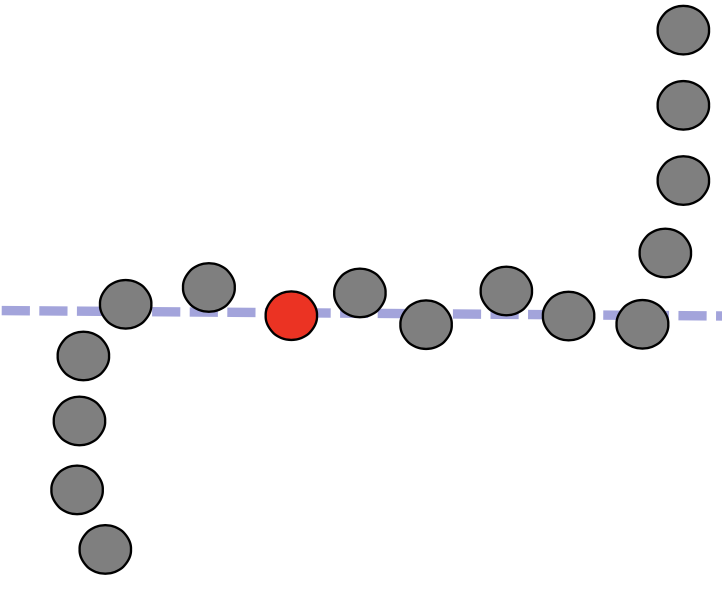
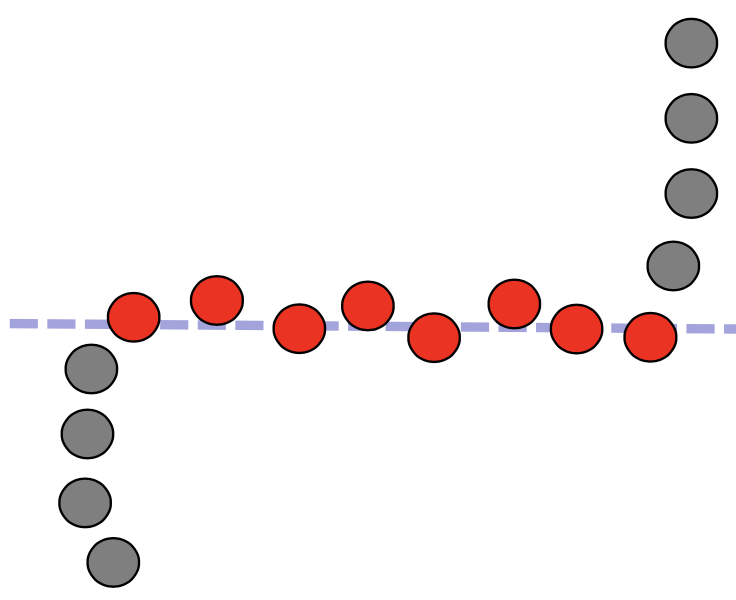
 

Fig. 1. Select a point Fig. 2. Iterate the process

and a primitive

hypothesis

Propagate to the neighbors if they are varified to the hypothesis using K-Nearest-Neighbor algorithm. Furthermore, iterate the process until no point is varified thr hypothesis anymore(Fig. 2).

Select one of the remaining point and primitive Hypothesis, and the same as step2, iterate the process(Fig. 3 and Fig. 4).

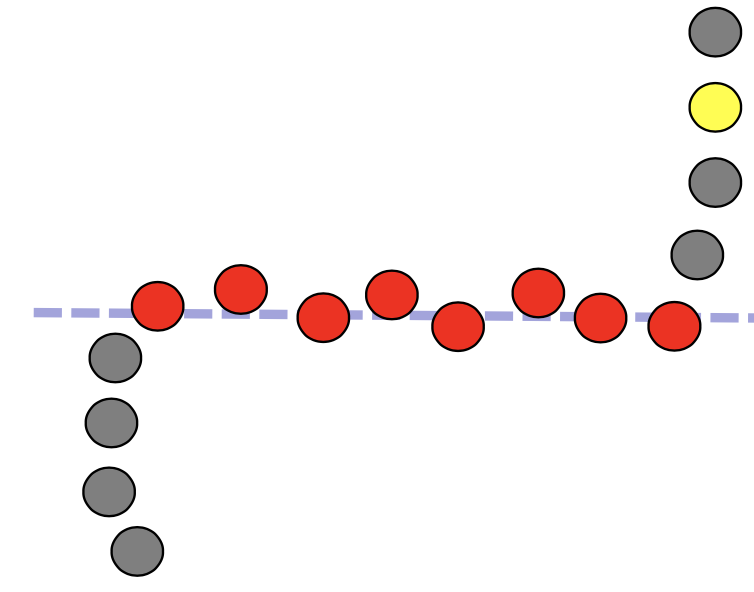
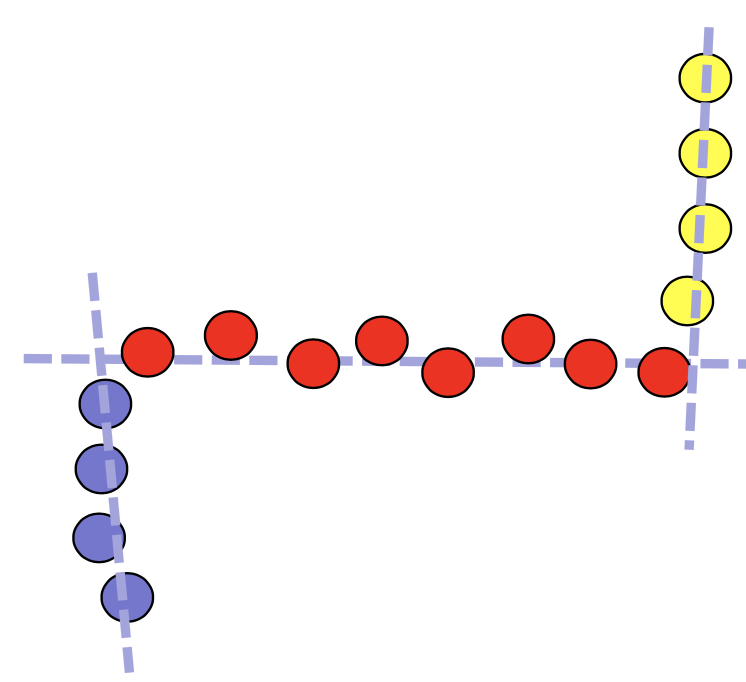
 

Fig. 3. Select a remaining Fig. 4. Iterate

point and a primitive

hypothesis

When applying region growing to cloud segmentation and shape detection or plane reconstruction, there are several issues need to be paid attention to. First and the foremost, we need to know the nearest neighbors, that’s why we use KNN to figure it out. Second, the primitives hypothesis has to be relevant when starting the growing. Moreover, the primitive hypothesis can also be updated during the growing. Last but not least, mind cleaning your cloud point set data, for the result might not to be optimal when there are tons of noisy points.

Various ways to apply region growing: using normals, using Euclidian distance, combine them both. As the pics show as follow.

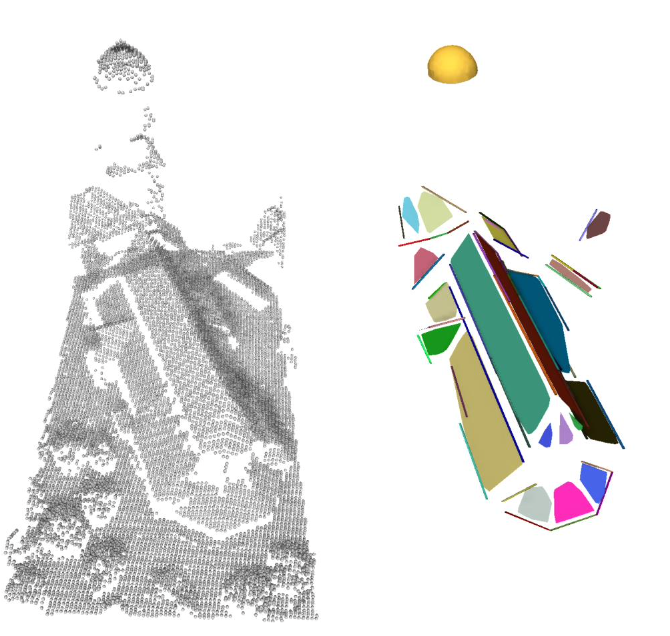
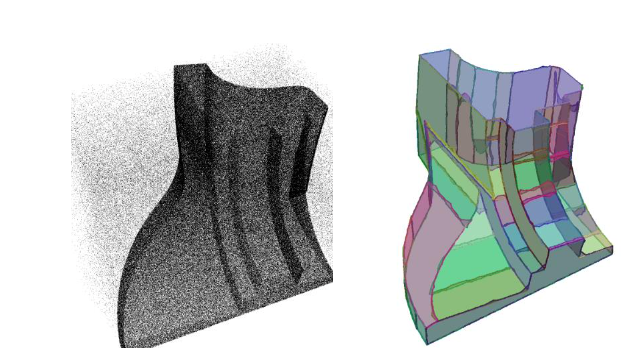
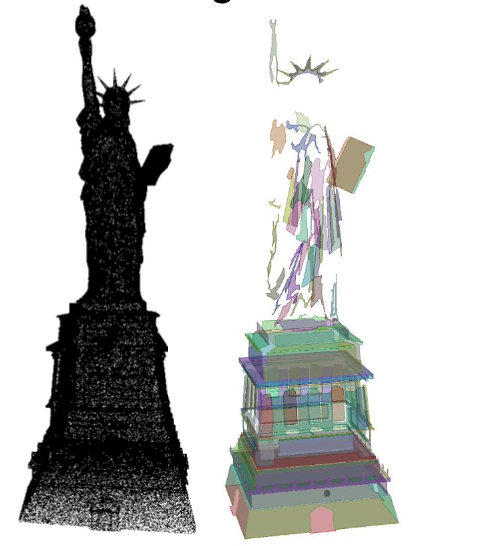
  

Fig. 5. Using normals Fig. 6. Using Euclidian Disdance Fig. 7. Using both

**2 Experiment Contents**

This experiment’s main target is to realize point cloud plane segmentation based on region growing algorithm. The goal is to input a ply file, which contains point cloud data. The way I see it, there are three models of the point cloud data. For instance, fandisk, cube\_lisse, cube\_distordu… etc. Fig. 1, Fig. 2, Fig. 3 shows these models.

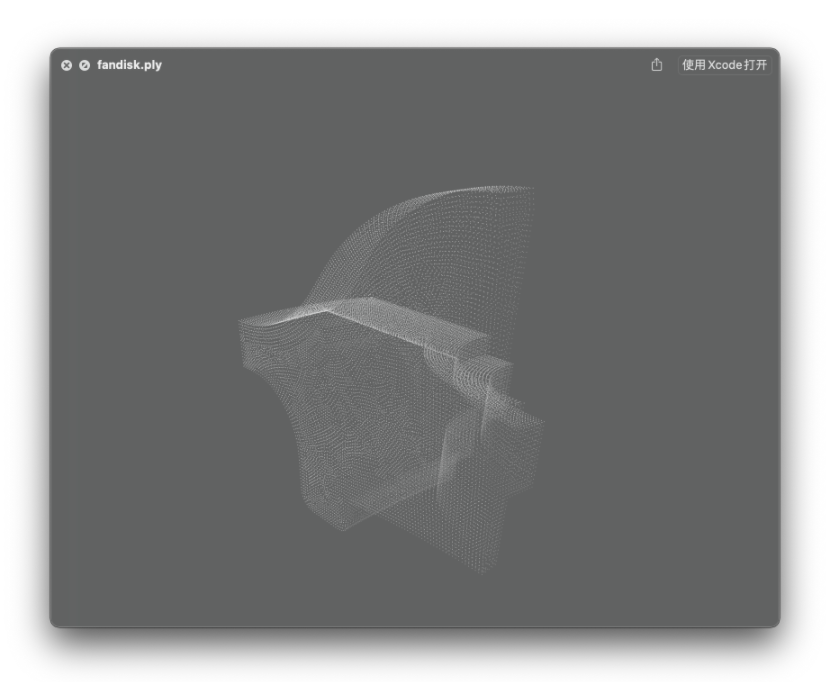
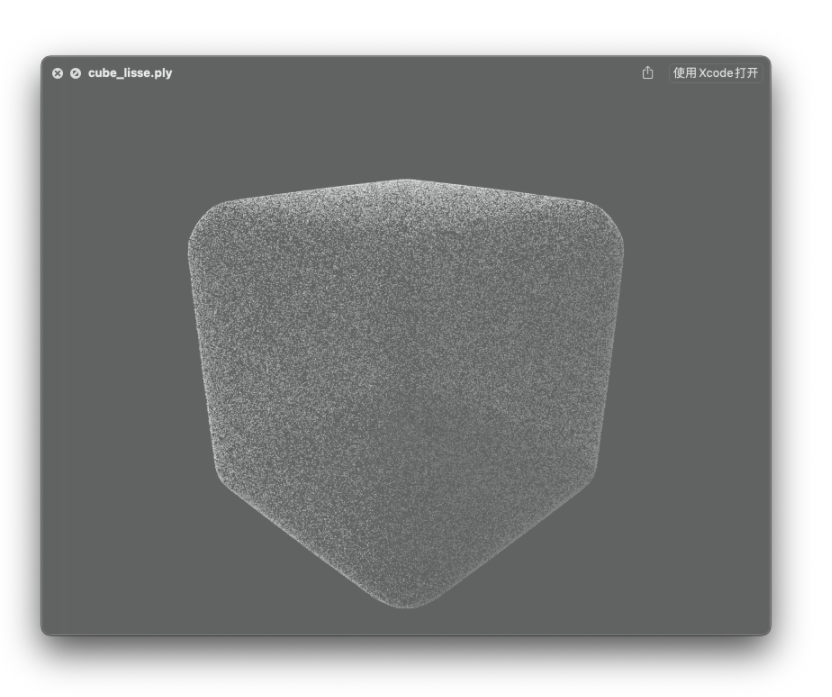
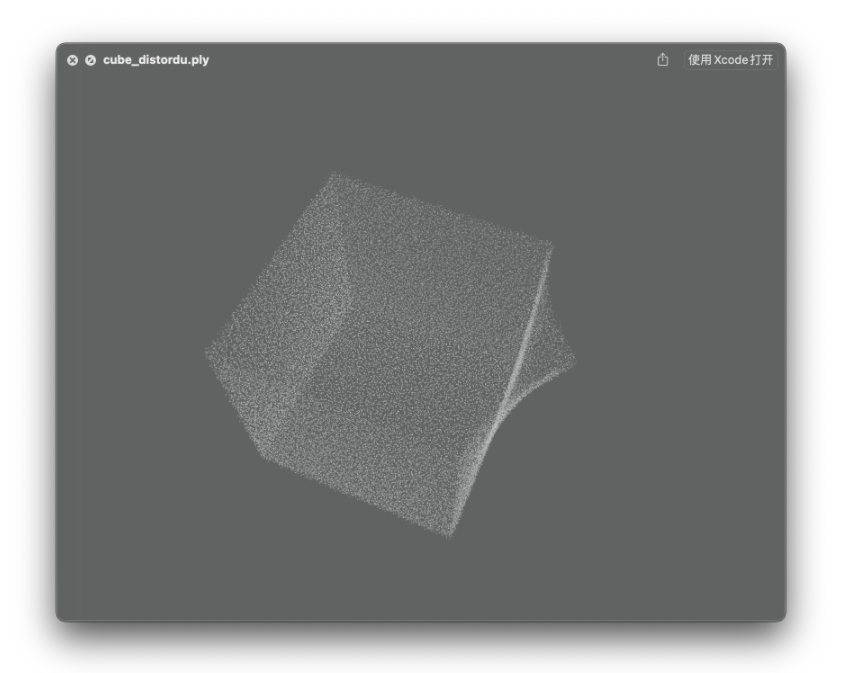
  

Fig. 8. fandisk.ply Fig. 9. Cube\_lisse.ply Fig. 10. Cube\_distordu.ply

The main target for this experiment is to realize this algorithm by completing the codes that are offered and run compilation on cmake by switching different two of the most important parameters to observe the outputs. Theoretically, the output should appear clusters that are segmented by varies of colors. Similar to the image that is offered by the tutor Hao fang in Fig. 4.

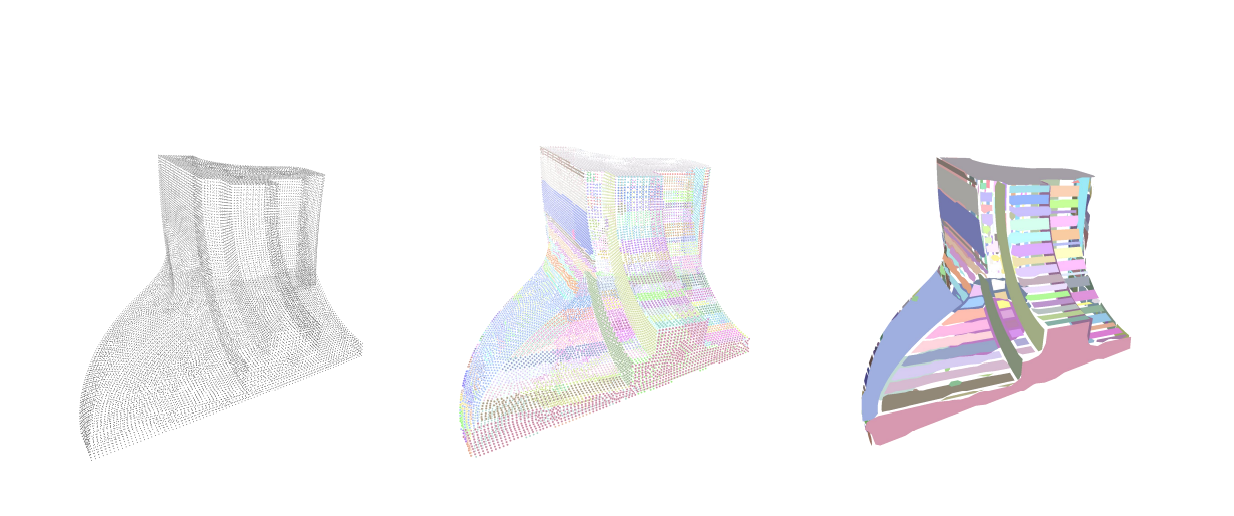


Fig. 11. Theoretical output

There are two main parameters: minimum number of points needed to fit the primitive, and fitting tolerance. According to the codes that are offered, I would like to call them epsilon and minimum\_num\_of\_inliers from now on. Learned from the slides of Assignment3, parameter epsilon ranges from 0.01 to 1 seems to affect segmentation planes in descending order, occ in ascending order, and time of processing in descending order. On the other side, the parameter minimum\_num\_of\_inliers ranges from 50 to 5000 seems to be affecting planes and occ in descending order, processing time in ascending order. Shows in Fig. 5.

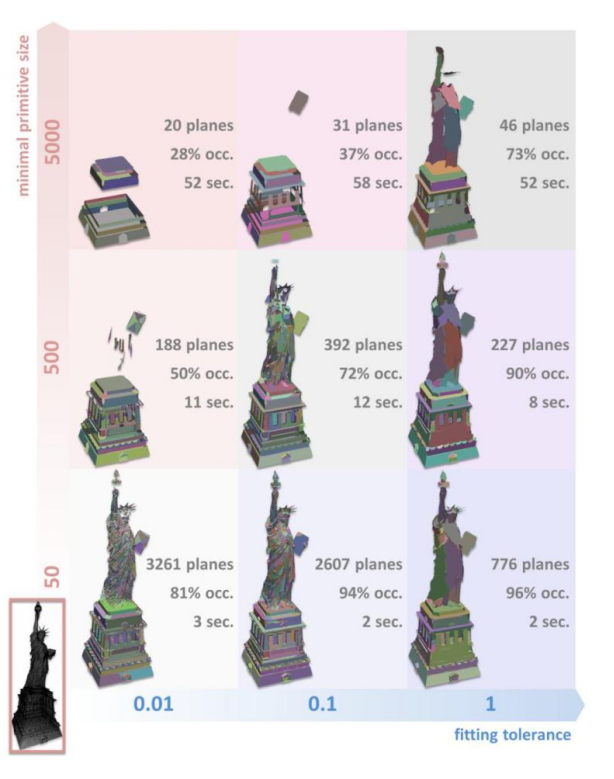


Fig. 12. Affections on different parameters.

**3 How I do it ?**

**Code’s Filling the blank**

After observing the codes in the file ‘cloud\_segmentation.h’, I noticed that we are missing a part that leads to iterate the process after we select a point and a primitive hypothesis.

Learning from a reference repository uploaded on the github which is copyrighted by INRIA Sophia-Antipolis (France), I completed the code of the missing part. Next step, I am going to compile this program and try different parameters and see what exectly affect the shapes. (armabon/UE4\_CGAL/ThirdParty/CGAL/includes/CGAL/regularize\_planes.h).

The part of the missing codes is completed and shown as follows:

// TO START FROM HERE (YOU)

//-----------------------

for (std::vector<int>::iterator it = index\_container\_former\_ring.begin(); it != index\_container\_former\_ring.end() ; ++it)

{

vector<int> neighbors = spherical\_neighborhood[\*it];

    for (auto curr\_select\_point : neighbors)

    {

       if (HPS[curr\_select\_point].primitive\_index != -1)

          continue;

       if (HPS[curr\_select\_point].normal \* normal\_seed >= epsilon)

       {

        HPS[curr\_select\_point].primitive\_index = class\_index;

          index\_container\_current\_ring.push\_back(curr\_select\_point);

        }

     }

}

if (!index\_container\_current\_ring.empty())

propagation = true;

//-----------------------

**Compile and run the codes**

In this part, I will conpile this program with cmake and run it with different combinations of parameters.

cd 3D\_VISION\_ASSIGNMENTS

cd Assignment\_2

mkdir build

cd build

cmake ..

make

./RegionGrowing ../cube\_lisse.ply

or ./RegionGrowing ../cube\_disortu.ply

or ./RegionGrowing ../fandisk.ply

Give value for epsilon: (range 0.01 to 2)

Give minimal number of inliers: (range 1 to 10000)

Region growing

-> xxx primitives

**Results**：

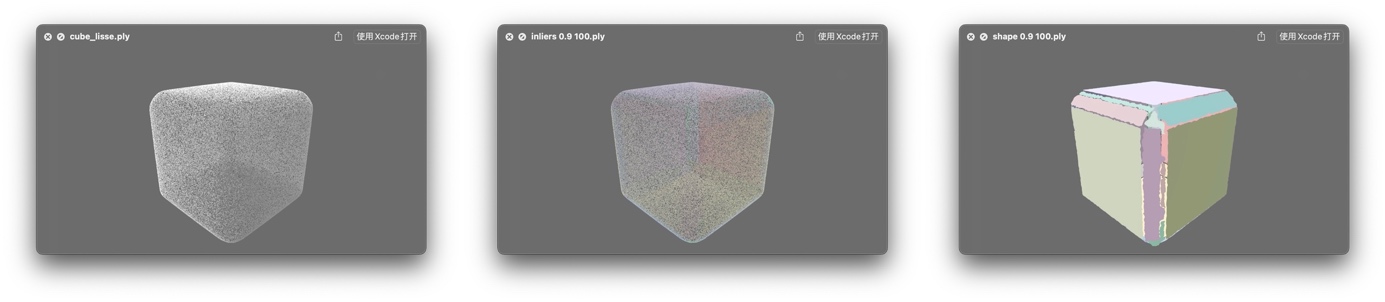


Fig. 13. Result of cube\_lisse.py using eposilon 0.9 and MNI 100

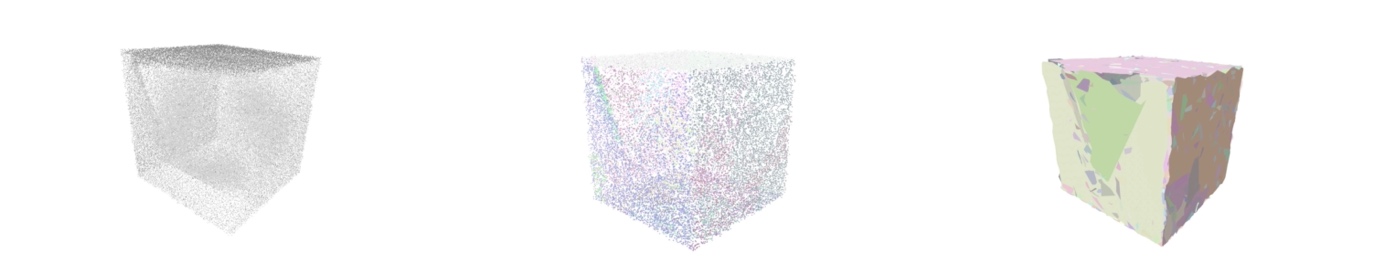


Fig. 14. Result of cube\_disortu.py using eposilon 0.93 and MNI 1

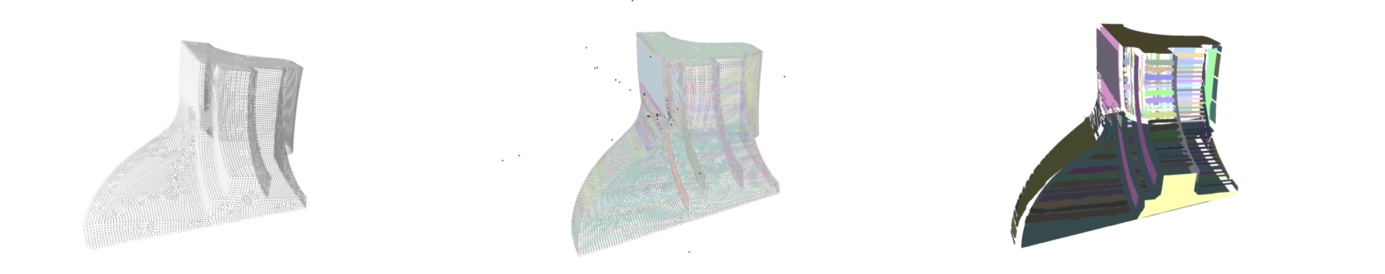


Fig. 15. Result of fandisk.py using eposilon 0.999 and MNI 1

**4 Analysis**

In this part I will analysis the result of region growing according to the graghs show that affections due to different paramaters.

After experimenting over 200 pairs of parameters, ranging epsion from 0.01 to 2, minimal number of inliers from 1 to 10000. In one hand, I found that the parameter epsilon seems to affect the distance from poonts to point cloud set’s suface or normals. On the other hand, parameter minimal number of inliers is affecting the size of point clusters to be involving into the process of region growing. Some pair-ups sometimes lead to surpeise results, but sometimes devasdating. It appears that it depends on the point cloud model’s number of surface and type of surface’s connection ways. Here are some try out parameter pairs, I draw an X-Y analysis tabel to illustrate them as is shown below.

**4.1 cube\_lisse**

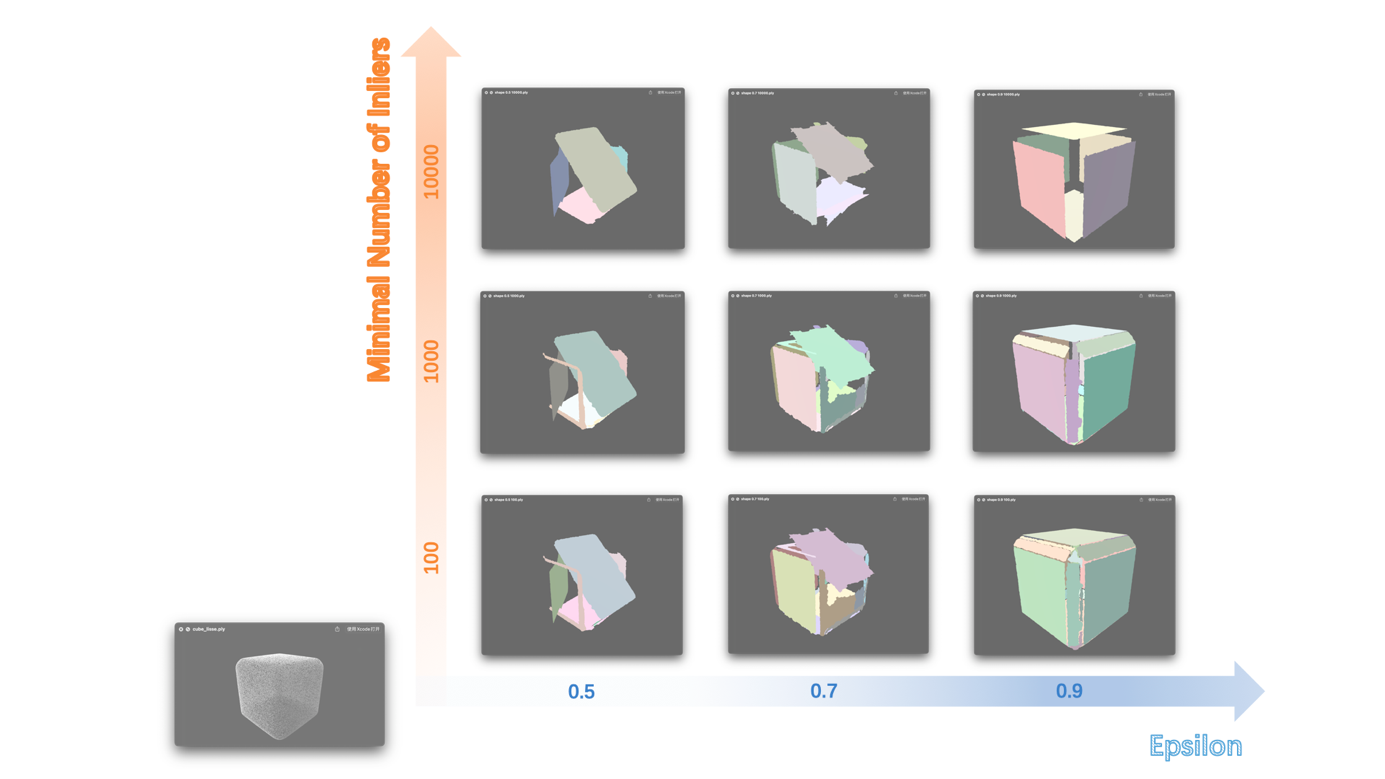
****

Fig. 16. Affections on different parameters on cube\_lisse

**4.2 cube\_distordu**

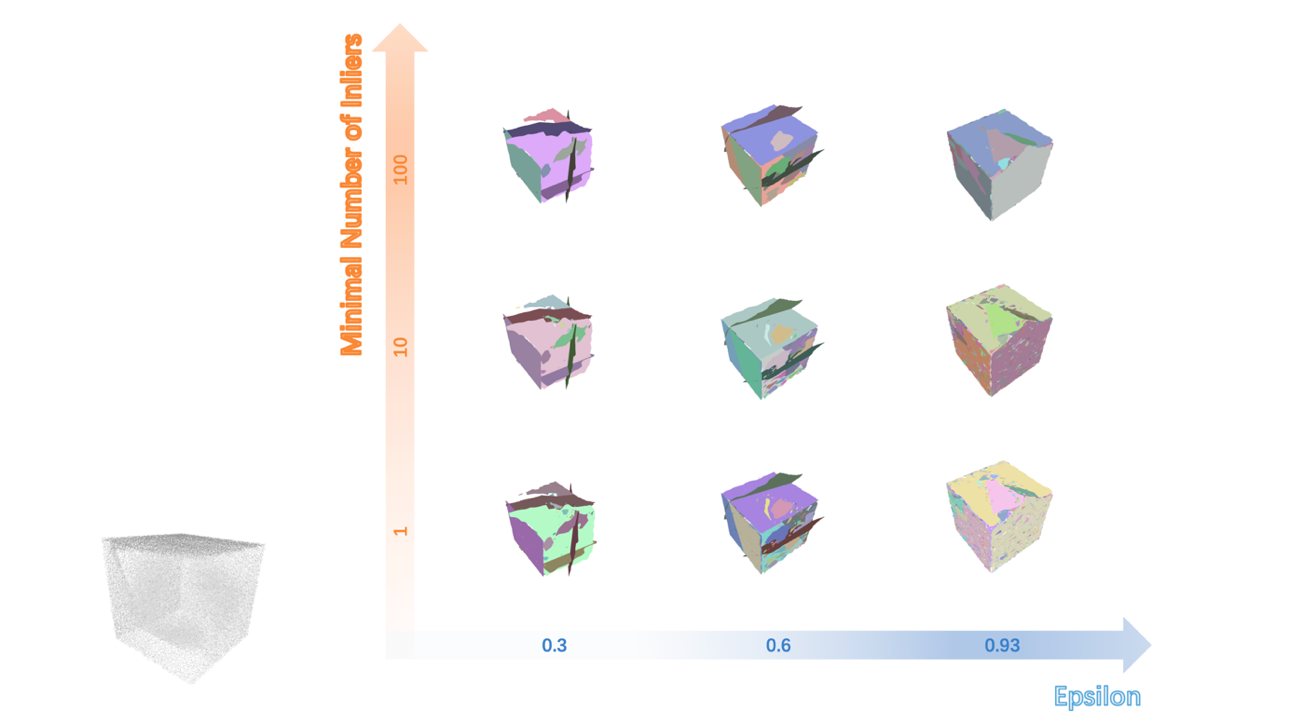
****

Fig. 17. Affections on different parameters on cube\_disortu

**4.3 fandisk**

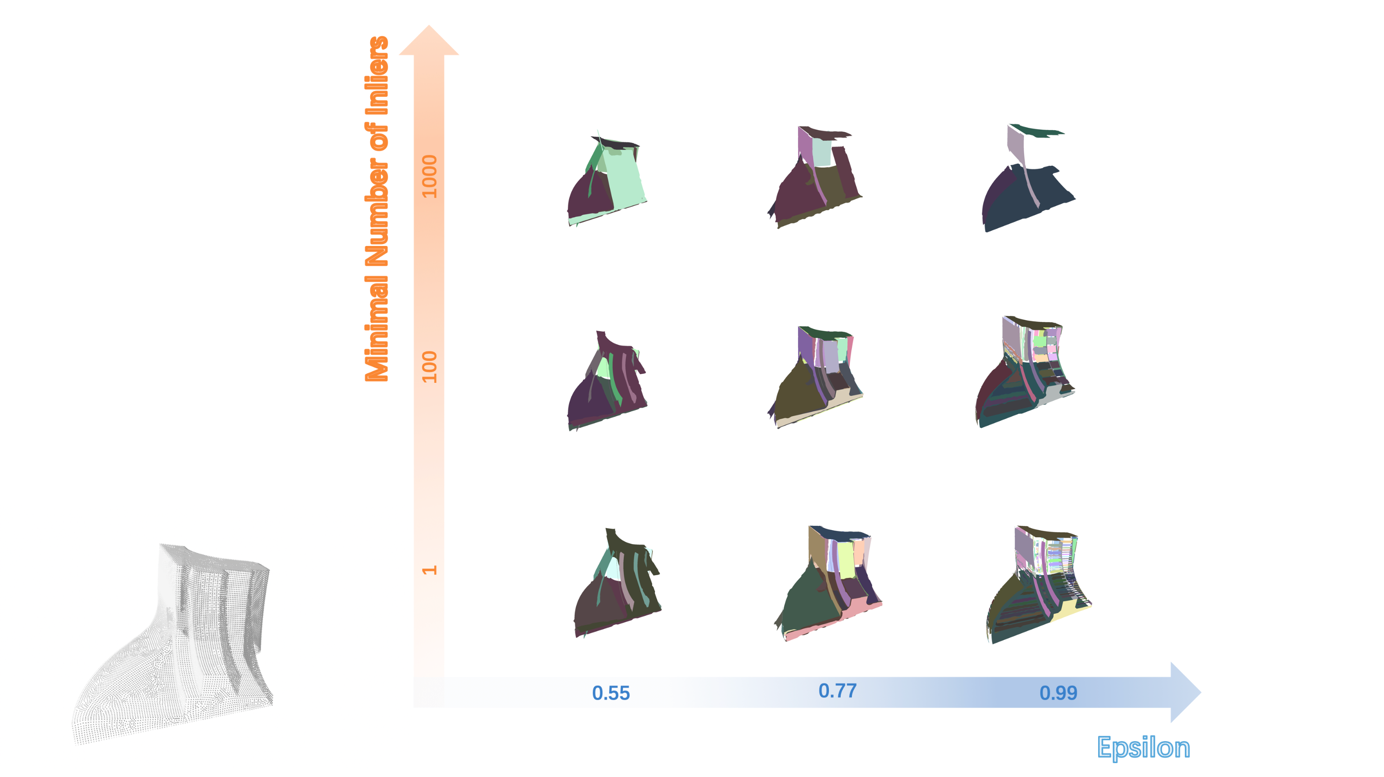
****

Fig. 18. Affections on different parameters on fandisk