TP5

Dany MOSTEFAI
Hady CHITER

February 2025

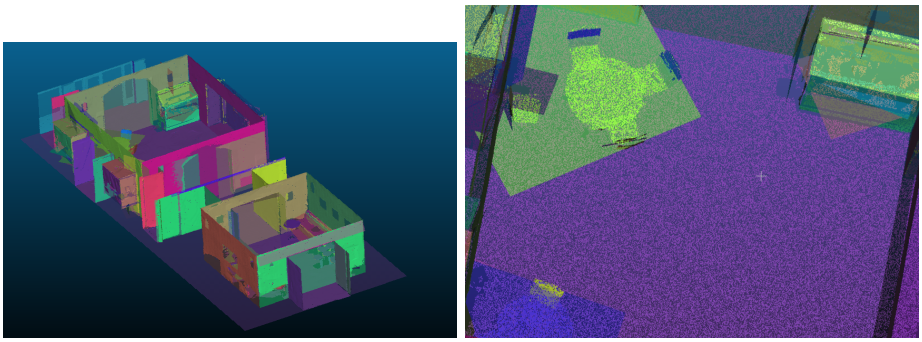
1 Question 1

Understanding the impacts of hyperparameters is needed to get a good segmentation.

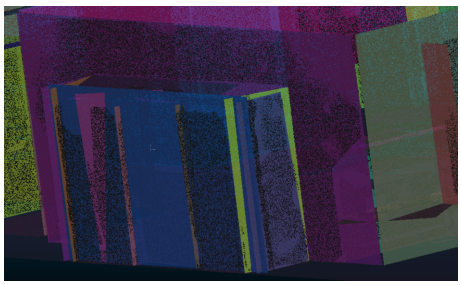
There are 4 hyperparameters we should modify:

- Min support points per primitive : the minimum number of points in a plane. For a low value, it will detect more specific and small shape, but increase the risk of false positives. For a high value, we obtain robust shapes, but we can overlook the smaller shapes.
- Max distance to primitive : this is the distance to the plane. Low value increases precision (We have less spreaded planes around their width).
- Sampling resolution : distance between neighboring points in the data. Low value increases the precision and sensitivity to noise.
- Max normal detection : maximum deviation from the ideal shape normal. Low values force to have points well aligned, but can some real world imperfections.

Figure 1 shows the segmentation with based parameters. While it looks good from afar 1(a), we truly view the limits of those parameters when looking into details. Figure 1(b) shows the ground of the house, where we can clearly see many plans that shouldn't exist (for instance, it created a plan for the table that is very large, and one cutting the clost in two). Furthermore, figure 1(c) shows how we get way too many plans for the simple front door.



(a) Global view (b) View of the ground

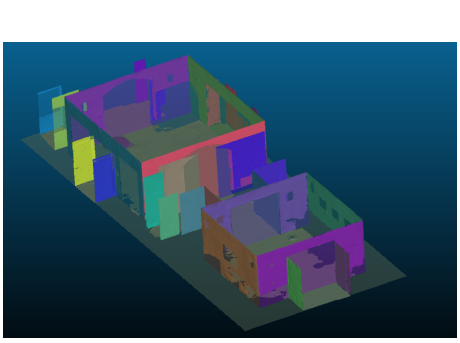


(c) View of the front door

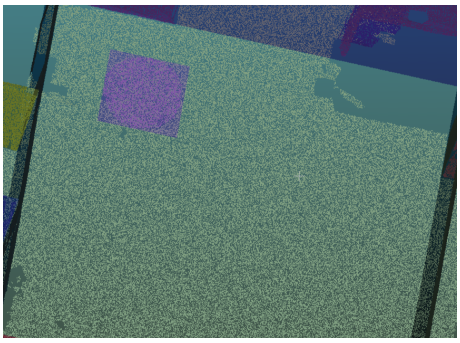
Figure 1: Obtained RANSAC segmentation with base parameters.

So our actual parameters fit too much noise, we would like robust and not too many plans.

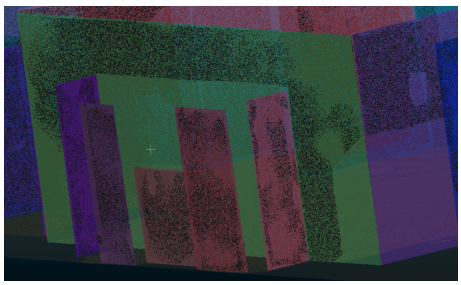
Figure 2 shows the output of the segmentation with our tuned parameters, where we obtain clean and well defined plans.



(a) Global view



(b) View of the ground



(c) View of the front door

Figure 2: Obtained RANSAC segmentation with tuned parameters.

Hyperparameters:

- Min support points per primitive = 10000
- Max distance to primitive = 0.1
- Sampling resolution = 0.2
- Max normal deviation = 12

2 Question 2

3 shows what has yielded our ransac algorithm. The red plan captures the whole ground, while the blue shape is not even a plane. We would like to have the second largest plan as the blue plane.

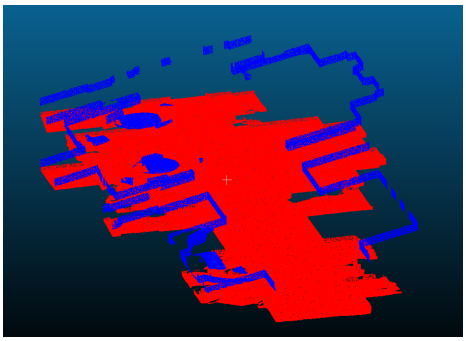


Figure 3: Segmentation with our RANSAC algorithm

This problem comes from the fact that we have a weak definition of a plan in our algorithm : we take three points, compute the plan and their points by using the distances, and count the number of points. But we never consider the normal of the plan of the points, then, it is logical that the algorithm will choose the slice containing the highest number of points, whatever the shape of this slice.

3 Question 3

We choose to run Ransac on "Lille_street_small.ply" (Figure 4).

The surfaces are nice, but there are some artifacts (Take for instance the dark purple plan on the front of the building).

Hyperparameters :

- Min support points per primitive = 500
- Max distance to primitive = 0.1
- Sampling resolution = 0.2
- Max normal deviation = 12

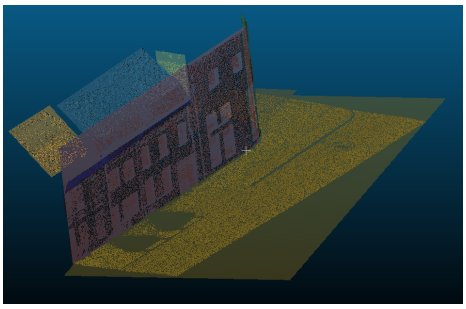


Figure 4: Visualization of the surfaces on the Lille point clouds.

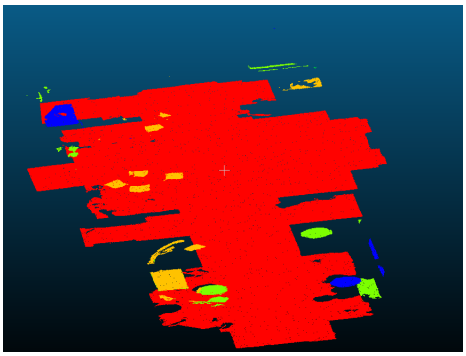


Figure 5: RANSAC with normals

4 Question 4

To remove the problem encountered in question 2, we should add a constraint on the normals of the points. Currently, we take the distance as a way to distinguish the points that are in the plan and out of the plan. The idea is to compute the normals of all the points, store them in a file (to avoid recomputing those each time), each time we consider a plan, we discard the points that have a normal that deviates too much from the normal of the plan.

Figure 5 shows the output of this new algorithm. We notice that we do not have any more those weird contours, however, the plans detected are very sparse.