TP 1 : Compte Rendu

Vincent HENRIC – Emma DUCOS
NPM3D - January 16th, 2020

A. Point clouds manipulations

Question 1 (2 point): Show a screenshot of the original bunny and the transformed bunny together. Pay attention to the appearance of the point cloud (activating EDL with perspective projection generally gives better visualizations). You should obtain something looking like figure 1.

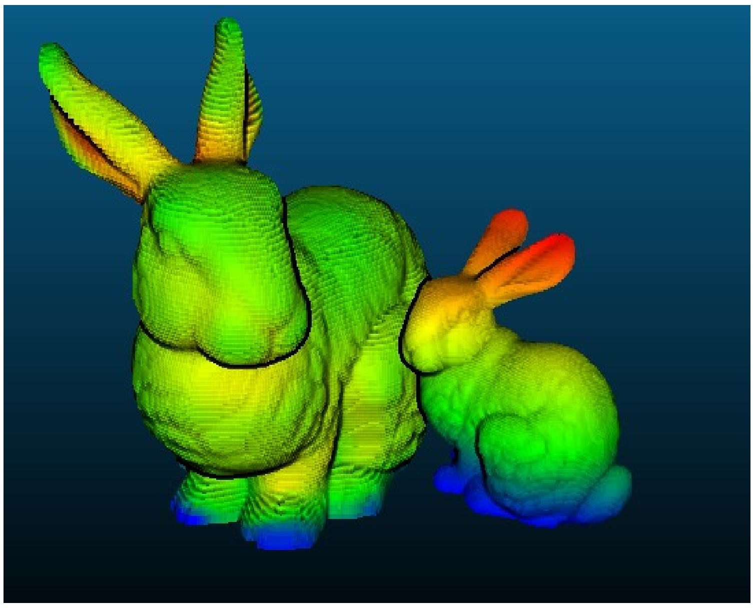


Figure 1 : screenshot of our bunnies

B. Structures and neighborhoods

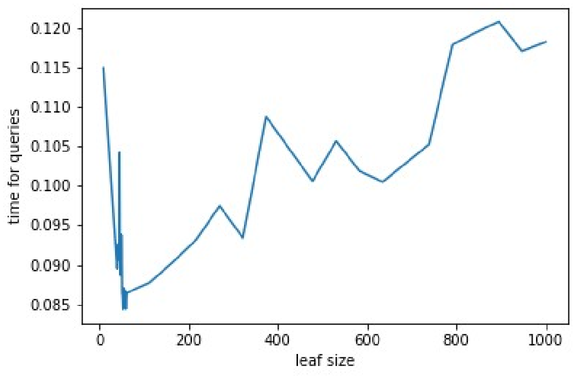
a. The concept of neighborhoods

Question 2 (1 points): Try to search the neighborhoods for 10 queries with both methods. Report the time spent. How long would it take to search the neighborhoods for all points in the cloud ?

It would take as much as 23 hours for the spheric method, and 261 hours for the k-nearest neighbors, while using the « argsort » fonction for sorting the values for the knn method. We could get better time performance if we replaced this function with the similar « argpartition » : it only sorts the first k elements, which is enough if we want k neighbors. With this function, the computation for knn comes down to 34 hours.

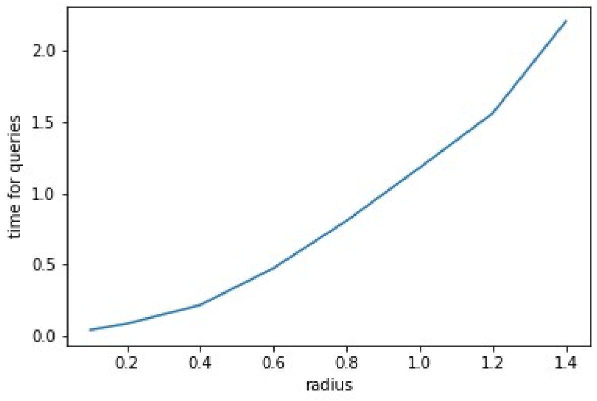
b. Hierarchical structures

<u>Question 3 (2 points):</u> which leaf size allows the fastest spherical neighborhoods search ? In your opinion, why the optimal leaf size is not 1 ?



We did a grid seach on a range from 1 to 1000 for the leaf size. For each leaf size, we average the time for 5 iterations of 1000 queries each to reduce variance (for computation purposes we did not try more iterations). A leaf size of 54 allows the fastest spherical neighborhoods search (of course there is still some noise, but the plot shows clearly the order of magnitude is fine).
A leaf size of one would mean that all the points would be in a leaf of the tree, which would make the search computationally expensive in time and memory space. As the leaf of the queried point contains only one point, we will have to go up in the tree the biggest number of times. In short, we will have a tree with the biggest depth, and need to do long traversals (downward traversal to identify the leaf, and upward to group with enough other leaves).

<u>Question 4 (3 points):</u> plot the timings obtained with KDTree as a function of radius. How does timing evolve with radius ? How long would it take now to search 20cm neighborhoods for all points in the cloud ?



Here timing is growing along with the radius of the neighborhood. As on average computing 1000 KDTree takes 0,115 seconds, it would take, for the whole point cloud, and with a radius of 20 cm, 5 minutes and 45 seconds to compute.

C. Subsampling methods

<u>Question 5 (2 points):</u> Show screenshots of the decimated point cloud and the grid subsampled point cloud in color. Comment on the advantages and drawbacks of each method.

The decimated cloud method is the easiest to implement but gives an unordered version of the point cloud. The initial dataset had points presenting a certain structure, here it is not the case (the image is much more « blurry »). On average, we keep similar points densities. But we also introduce noise because we do not really control what points are going to be removed ; with bad luck, all the points representing a particular object may disappear. In short : it is fast to implement, keeps point density, but is an undeterministic process (in the sense that we do not have any information on the order of the points in our original list) that can lead to noise/variance in the sampled representation.

The subsampled point cloud is more complex and computationaly demanding. It keeps a structure more representative, organized, without much noise. However, it might simplify regions where the density of points was initially high.

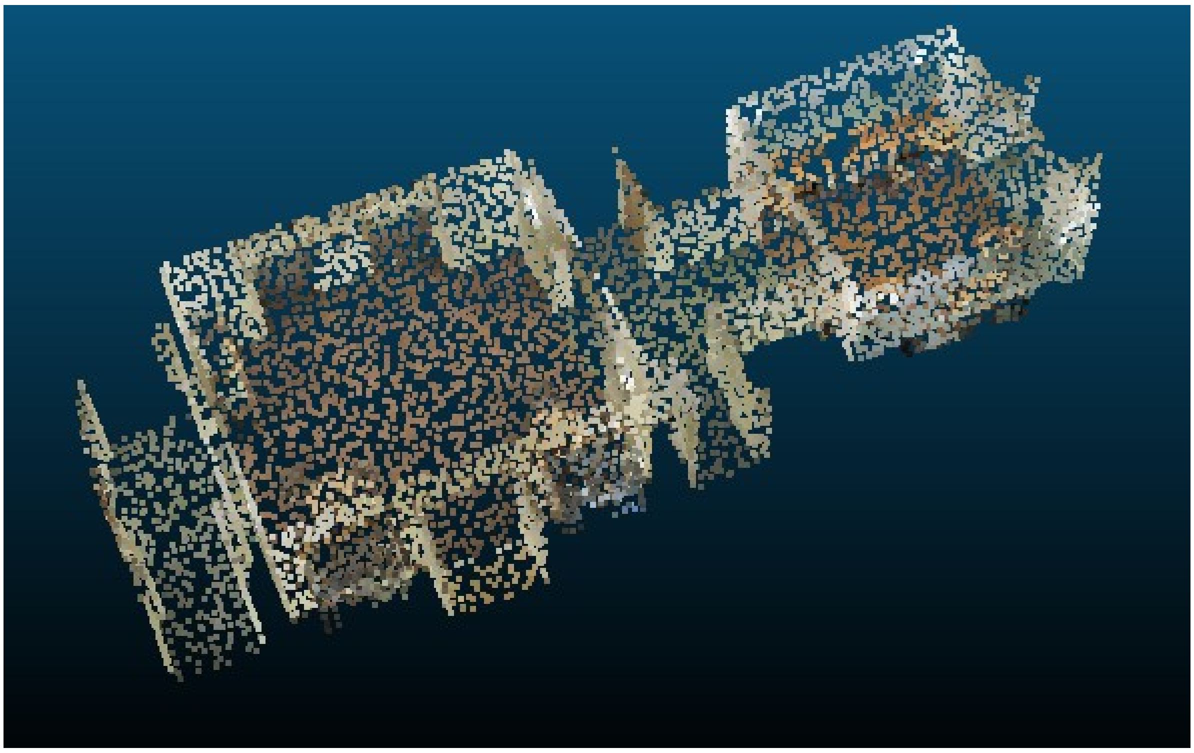


Illustration 1: Screenshot of the decimated point cloud

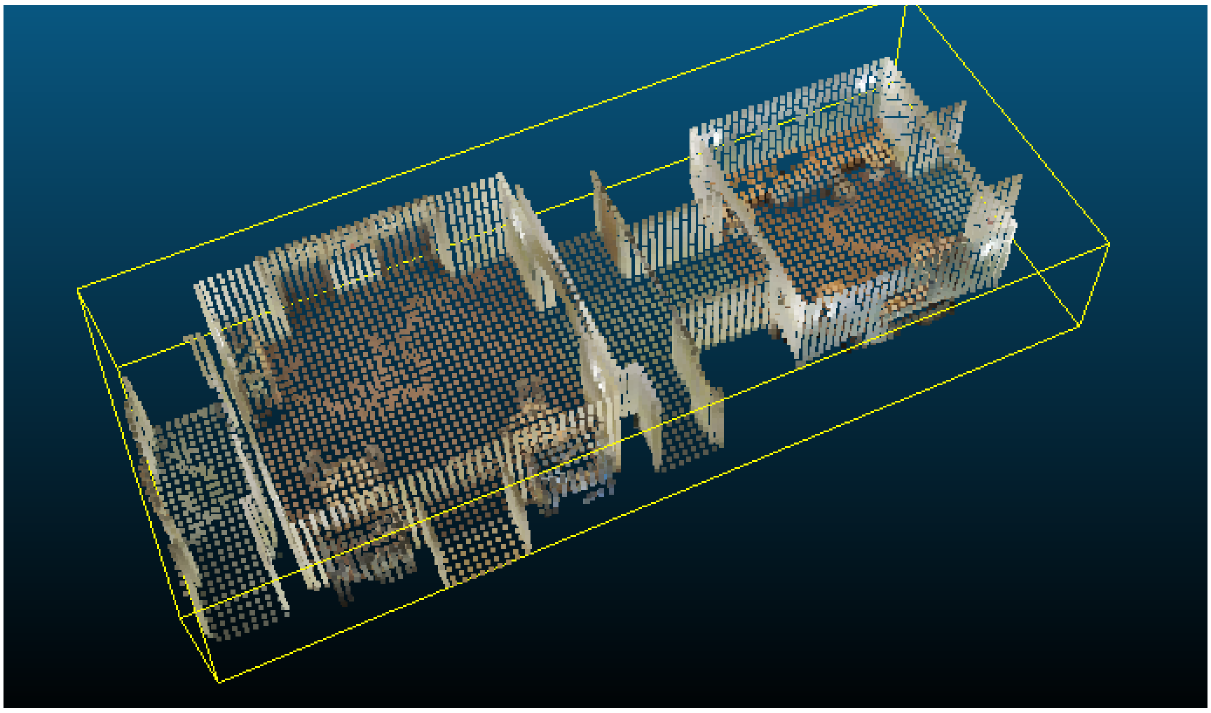


Illustration 2: Screenshot of the subsampled point cloud

<u>Question Bonus 1 :</u> **Show a screenshot of the grid subsampled point cloud with labels.**

If the label is a real number, we can simply average, as we do for colors. If it is a categorical data, then we take the mode on the voxel. If the categories were to be ordered, we could have thought of taking the median of the labels in the voxel. In the plot, the different labels have different colors.

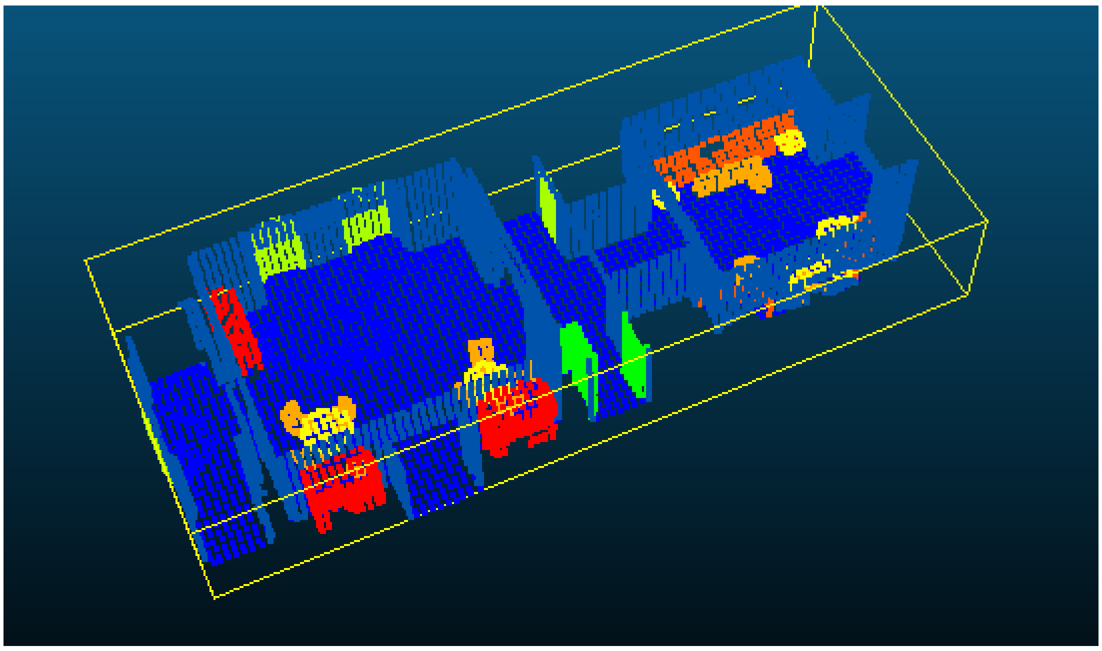


Illustration 3: Screenshot of the labelled point cloud