NPM3D - TP3

Aubin TCHOÏ

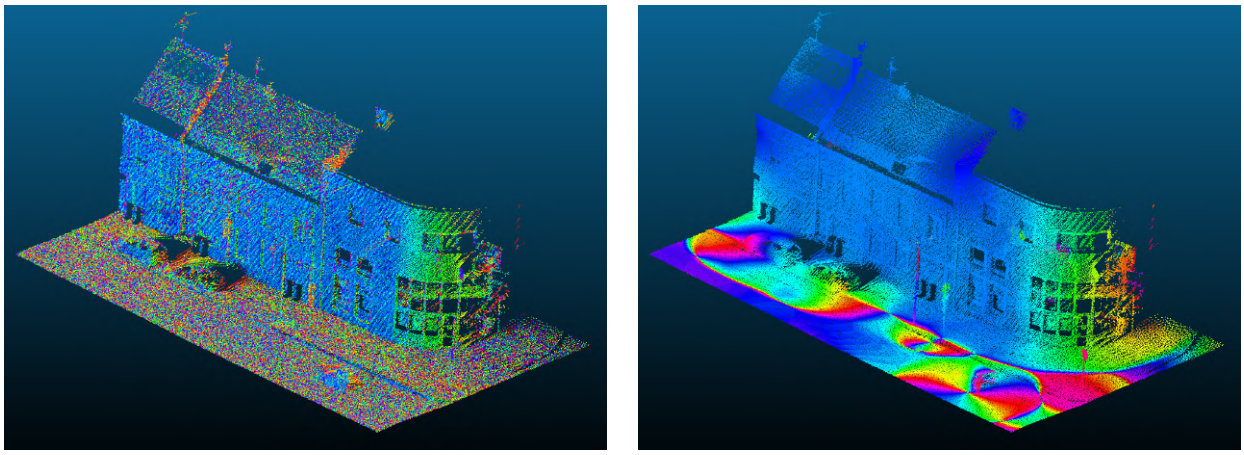
January 2023

Question 1

When computing normals on a point cloud with spherical neighborhoods, if the search radius is too small the neighborhoods associated with each point will only contain a few points and the noise on the positions of these points will induce a noise on the direction of the normal computed. If there were more points in each neighborhood, the noise would be averaged and the normals would be less noisy. This is highlighted by the figure below on the left.

If we choose a radius that is too big, local information may disappear. For instance in the figure below on the right the details around the windows disappear and the normals computed have the same direction as the ones of the points on the wall. In this example this is caused by the fact that the neighborhoods of the points in the borders of the windows include mostly points that are on the wall since we took a radius too big. Therefore a radius too big leads to uninformative normals and to the disappearance of local information.

The screenshots below show the normals computed on point cloud LILLE_STREET_SMALL.PLY for two different values or the radius that is used in the neighborhood search.



a) b)

Figure 1: Results obtained with CloudCompare "Normals" tool for various neighborhood radii a) radius of 0.1 m b) radius of 2 m

Question 2

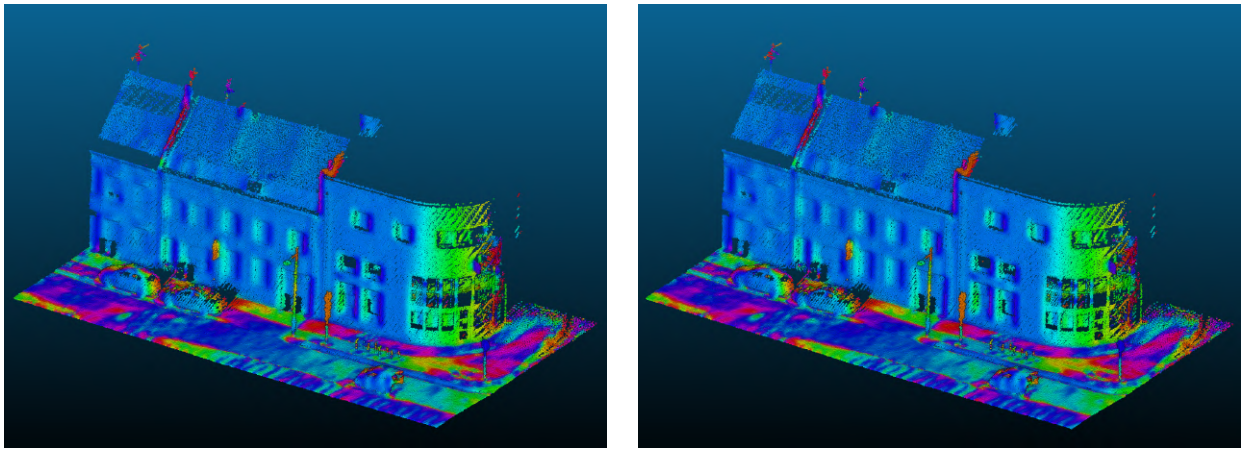
As highlighted above, there is a trade-off to consider when choosing the size of the neighborhood scale, since we want a neighborhood big enough to average the noise on the data, and small enough so that it correctly captures local information. I would say that the two scaling factors to consider here are the typical scale of the small details of the image; if you were to take a spherical neighborhood of radius bigger than some object then every detail on the normals around this object would disappear. The other factor to take into account would be the density of the point cloud, if the point cloud is very dense you can lower the search radius without risking being subject to noise since the neighborhoods will still be highly populated.

A systematic approach would consist in checking the optimal value of the least square problem of approximating the neighborhood by a plane, which is equal to the smallest eigenvalue of the covariance matrix. This value follows a chi-squared distribution if the neighborhood is indeed planar with Gaussian noise, and therefore we can determine whether or not to reject the hypothesis of planarity. If we do so in a significant portion (depending on the nature of the scene) of the point cloud it means that we chose neighborhoods that are too big.

Question 3

The normalized eigenvector associated to the smallest eigenvalue is the best approximation of the normal to the surface.

Below on the left is a screenshot of the normals computed with function `compute_local_PCA` converted as "Dip" scalar field in CloudCompare with a radius of 50 cm. On the right is a comparison with the result obtained previously using CloudCompare's "normals" tool with the same radius.



a) b)

Figure 2: Visualization of the normals implemented using
a) `compute_local_PCA` b) CloudCompare's "normals" tool

We obtain the exact same results, which validates our Python implementation.

Question 4

The figure below is a screenshot of the normals converted as "Dip" scalar field in CloudCompare computed with k-nearest-neighborhoods with $k = 30$.

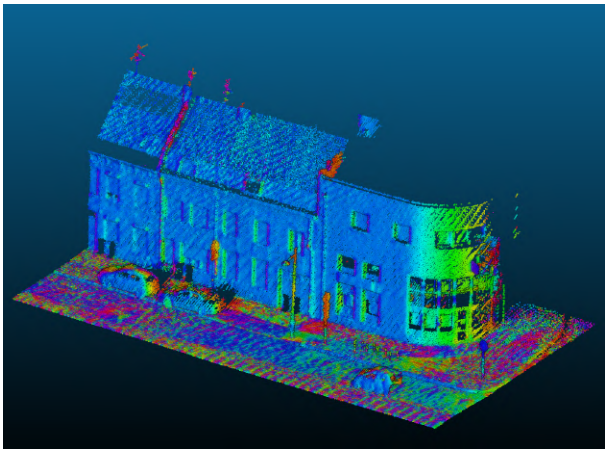


Figure 3: Visualization of the normals computed using knn-neighborhoods

The figure obtained is visually noisier, especially on the ground and on the roof. We also observe that the edges of the objects in the point cloud are much sharper, especially around the windows of the building, and that the result is very similar on the round part of the building on the right. This is due to the fact that the point cloud is less dense in this part and that the spherical neighborhoods then become similar to the knn ones. The other parts of the cloud have higher densities, which explains the noise because spherical neighborhoods then become a lot more populated than knn ones. The approach using knn neighborhoods can be seen as a spherical search with radii that vary depending on the density of the point cloud.

The histogram below shows the sizes of the neighborhoods obtained previously with a spherical search of radius 50 cm. The average neighborhood size is 171.5 and the standard deviation on its distribution is 51.2 with values varying between 1 and 339. These values show that with $k = 30$ we choose smaller neighborhoods in a vast majority of cases, and that the density of the point cloud is uneven, for LiDAR sensors it typically decreases with the distance.

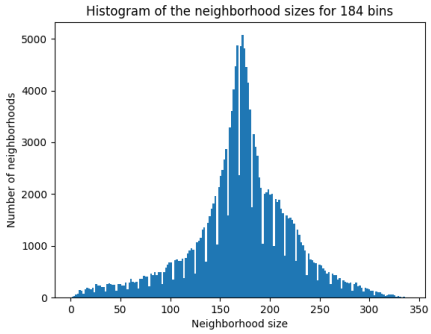
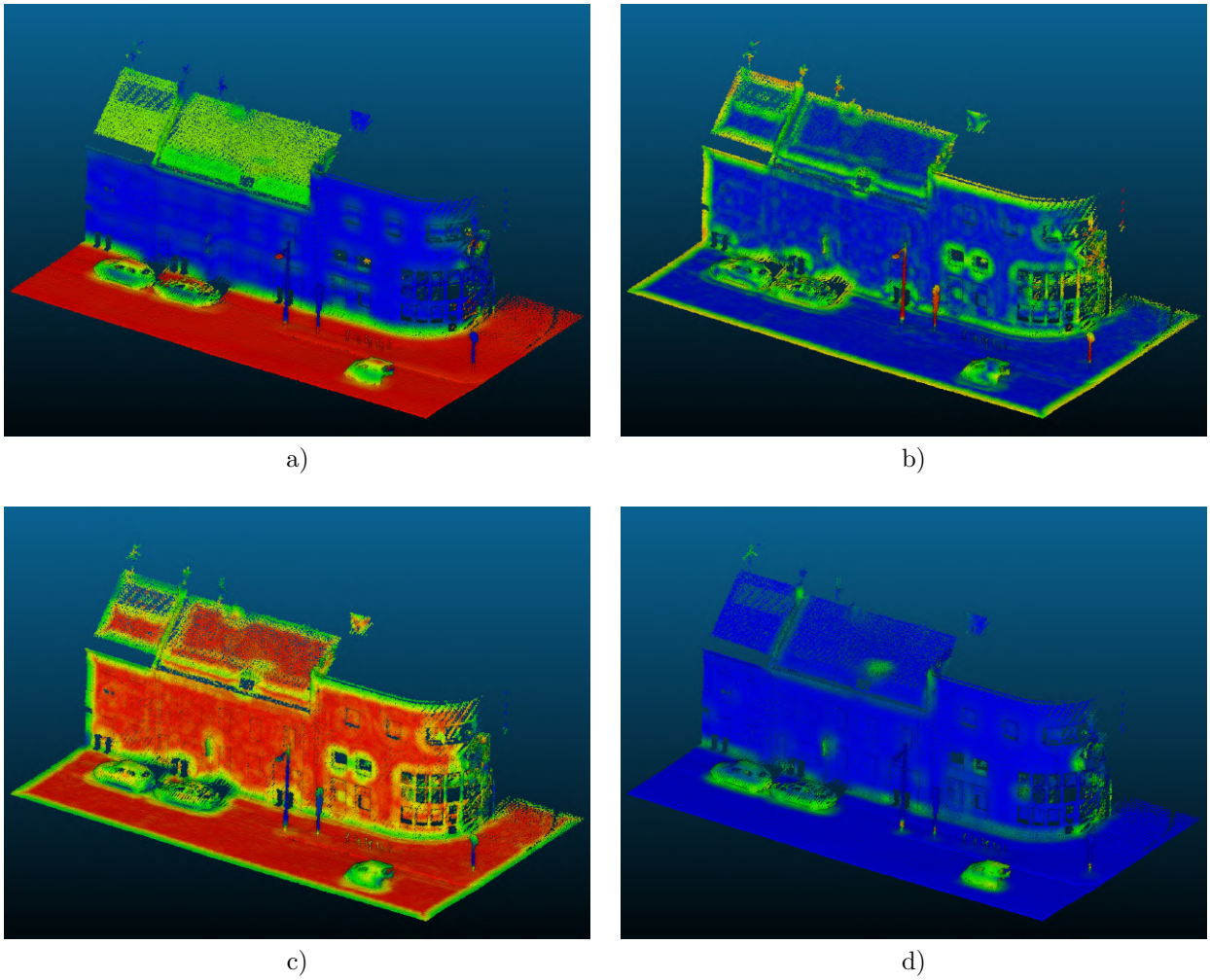


Figure 4: Histogram of the sizes of the neighborhood for the spherical search

Bonus question

The figures below show screenshots of the four features as scalar fields of the point cloud.



a) b) c) d)

Figure 5: a) Verticality b) Linearity c) Planarity d) Sphericity

The three eigenvalues of the covariance matrix describe how much the neighborhood expands on the orthogonal directions given by the eigenvectors.

If the linearity is close to 1 it means that $\lambda_1 \gg \lambda_2 \geq \lambda_3$, which happens when the point cloud is locally elongated around a single direction and compressed around the barycenter on the two other directions.

Planarity is the same for two directions instead of one, it is close to 1 when $\lambda_1 \approx \lambda_2 \gg \lambda_3$, which means that the point cloud is equally expanded around only two directions and flat on the remaining one.

Sphericity is close to 1 when all three eigenvalues are close, which is the case for a point cloud that is equally expanded in all three directions, for instance a sphere with even density has a sphericity of 1.