Universitat Politècnica de Catalunya (UPC)
Universitat Autònoma de Barcelona (UAB)
Universitat de Barcelona (UB)
Institut de Ciències Fotòniques (ICFO)

http://www.photonicsbcn.eu

Master in Photonics

MASTER THESIS WORK

Point cloud management techniques for a multihit ladar imaging camera system

David Camps Novi

Supervised by Dr. Santiago Royo, (CD6)

Presented on date 7th September 2016

Registered at

Escola Tècnica Superior d'Enginyeria de Telecomunicació de Barcelona

Point cloud management techniques for a multihit ladar imaging camera system

David Camps Novi
Centre for Sensors, Instruments and Systems Development (CD6), Universitat Politècnica de Catalunya (UPC), Rambla Sant Nebridi 10, 08222 Terrassa, Spain

E-mail: david_camps_93@hotmail.com

Abstract. With the incorporation of optical sensors into the machine vision technology, a full new field has emerged to revolutionize different technologies such as self-driving, 3D scanners and printers or virtual reality. However, new technologies come with new techniques and methodologies to manipulate them. Point Clouds were born as the data storage system and a collection of challenges came with them. One of these challenges consists in processing them in order to obtain the best description of the real world. Hence, it is necessary to have a tool to evaluate the quality of those Point Cloud in order to analyze their quality. In this MSc thesis we developed a mathematical approach for Point Cloud quality evaluation and implanted by Matlab. The full mathematical development as well as the structure of the code and the different tools used to acquire and manipulate Point Clouds are described and introduced along the thesis. A final analysis of the methodology showed there is still a lot of work to do. Several questions appeared and need to be solved in order to grow in this field.

Keywords: point clouds, time-of-flight, machine vision, image processing, surface normal vectors.

1. Introduction

1.1. Point Clouds and Machine Vision technology

As time passes, humans go further in the development of new technologies to make the world a better place. Recently, we have seen how the field of images made a transition from 2D to 3D. Even in the daily life, we can notice that 3D technology is around us in simple things such as the cinema or the photography.

However, 3D technology has gone further and reached other fields despite we cannot realize as easily. This is the case of the robotics technology; more specifically, the machine vision or computer vision technology, which is growing extremely fast. [1]

The robotic vision, or machine vision, provides imaging-based perception through 3D sensors. Over the past few decades, this technology evolved from simple range sensors such as sonars to complex systems such as LiDAR. This last method provides information of the real life through the so called point clouds, a high-quality 3D representation of the world. [2]

Point clouds represent the basic input 3D information for those machine vision perception systems. They are a discrete 3D representation of the real world through x, y and z coordinates, having its origin in the sensing device that acquired the data. Among many others, time-of-flight systems are very common as 3D optical sensors. They measure the time delay in which emitted light exits the sensor, hits a surface and returns to the device. [3]

1.2. The Point Cloud Library (PCL)

The PCL is a large scale, autonomous open project for n-D image processing. It provides a lot of useful tools to work with point clouds.

It is fully integrated with the Robot Operating System (ROS) and has been used and many projects of the robotics community [2].
Its architecture consists in fully template modern C++ and most of their internal procedures are detailed in several documentations of their webpage, being a very powerful source to understand the working mechanisms [4].
The existence of the PCL had a crucial role in this thesis because of the huge amount of detailed documentation that allowed us to understand the point clouds behavior as well as how to manage them through the different softwares and tools we used.

1.3. Point Cloud quality assessment
As well as in 2D images, point clouds also need a collection of methods and tools to evaluate and improve their quality. However, Point Cloud image quality assessment is not enough controlled yet. No methods are reported to be used for Point Cloud evaluation directly from the images. It is thus necessary to define new methods to deal with point cloud quality.

1.3.1. Point Feature Histograms and Surface Normal Vectors
Surface normal vectors are used in mathematics to define the orientation of planes. Hence, using them in the proper way we can define a surface (‘figure 1’). This is the idea behind the Point Feature Histograms, a technique reported in the PCL documentation from a PhD thesis [3].

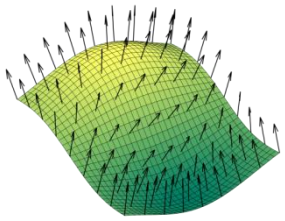
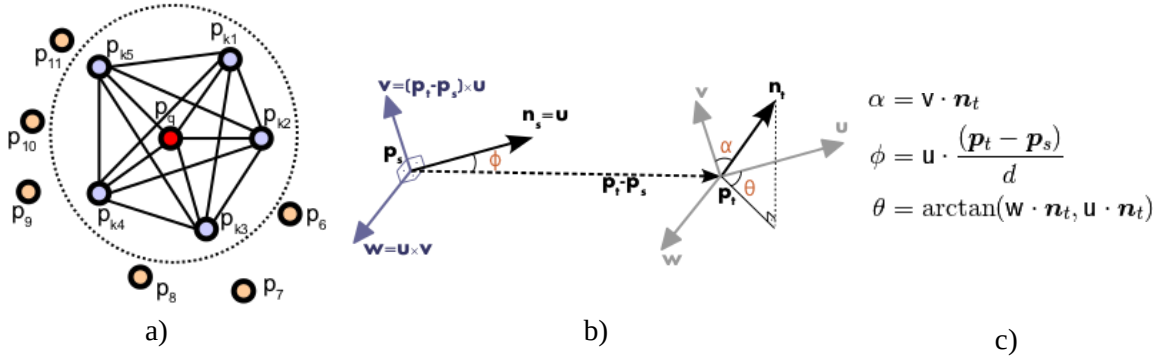


Figure 1. Representation of the surface normal vectors along a smooth surface. The orientation of the vectors as well as their local properties can describe the shape of the surface.

PCL refers to Point Feature Histograms (PFHs) as a tool to encode geometric properties of a point cloud through the local curvature in each point. Sampled surface variations over a point can be computed through the interactions with its neighbors and then give an overall geometry when going along the whole point cloud. [3]
PFHs work with the Darboux Frame, a new reference coordinate system defined for each pair point-neighbor (p_q-p_k) to compute the difference in their orientations (‘figure 2’).



a) b) c)

Figure 2. Schematic calculations for the PFHs [3]: a) selection of a query point p_q and its neighbors p_k, b) Darboux frame for each pair p_q-p_k, c) calculation of new angular variables from the Darboux frame.

Orientation differences are stored as information in different angles and distances to compute the final histograms (PFHs) that will give information on the geometry (‘figure 3’).

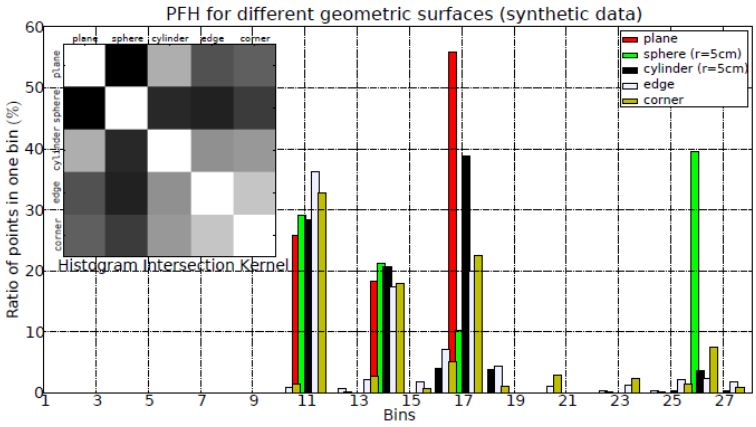


Figure 3. Point Feature Histogram for different geometries [3]. Each geometry have a certain pattern of columns.

2. Mathematical approach

The goal of this thesis is developing a method capable to evaluate quantitatively the quality of point clouds. To do it, we designed a mathematical approach based on the PFHs.

Previous work done by Rusu [3] found that Gaussian noise affected the PFHs in such a way that noisy point clouds have some columns that are absent in clean point clouds (‘figure 4’).

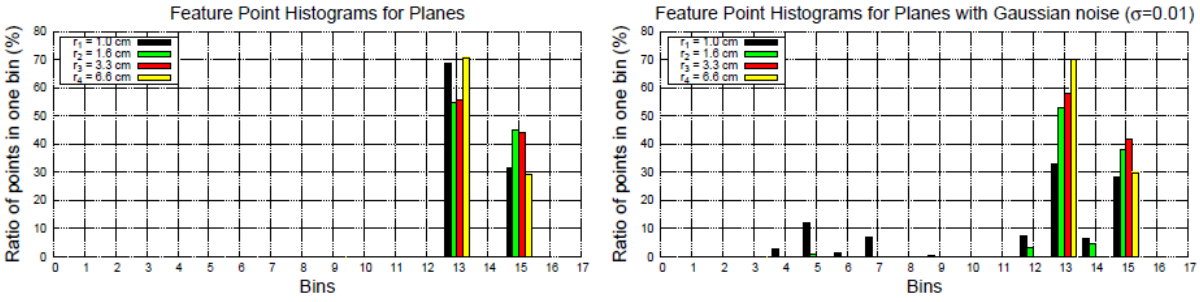


Figure 4. Point feature histograms whose neighbors have been selected from different distances to the query point so from different number of neighbors [3]: a) without noise, b) with Gaussian noise. In figure 4b, the presence of Gaussian noise can be identified by the presence of new columns absent on 4a, especially when the radius is lower and thus the number of neighbors (black and green columns).

Since the presence of noise can be identified through the PFHs, surface normal vectors might be affected. And the reason is that smooth regions of point clouds have a constant angle between adjacent normal vectors, while sharp structures such as corners or high-noise regions not (‘figure 5’). Hence, a method that computes the ratio of regions with smooth regions should give information on the whole point cloud quality. And this is what we wanted to calculate.

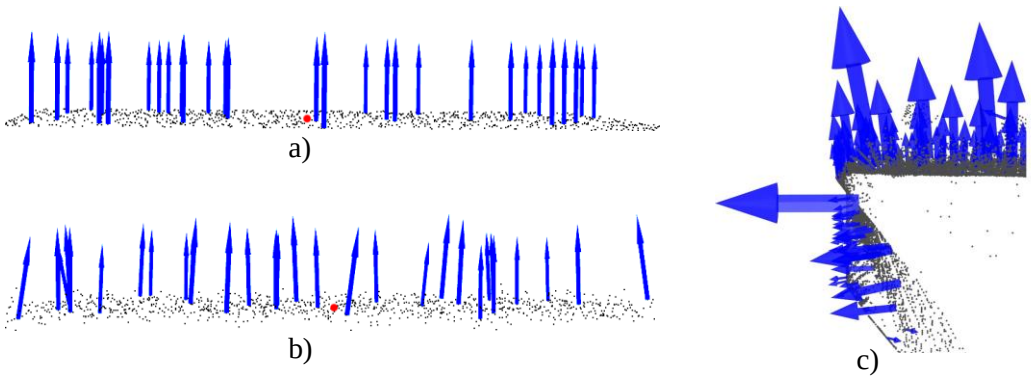


Figure 5. Representation of surface normal vectors in different cases [3]: a) smooth surface, b) noisy surface, c) edge. The orientation of the vectors can distinguish a smooth surface from the other two.

As Rusu, we defined a neighborhood for each point ('figure 2a') and computed all their normal vectors. Then, we calculated the angle between each pair p_q-p_k for each neighborhood. This angle is not one of the angles defined from the Darboux frame ('figure 2c') but a new one.
Since in any smooth surface the angles between adjacent normal vectors are constant, the angle between each pair p_q-p_k must be equal for the whole neighborhood.
In order to make these calculations we used Matlab. Data preparation was made through several softwares. We are going to explain all those techniques along next sections.

3. Point Cloud quality evaluation

The evaluation of point clouds was a complex task that required the use of several programs in order to prepare the data for further calculations ('figure 6').

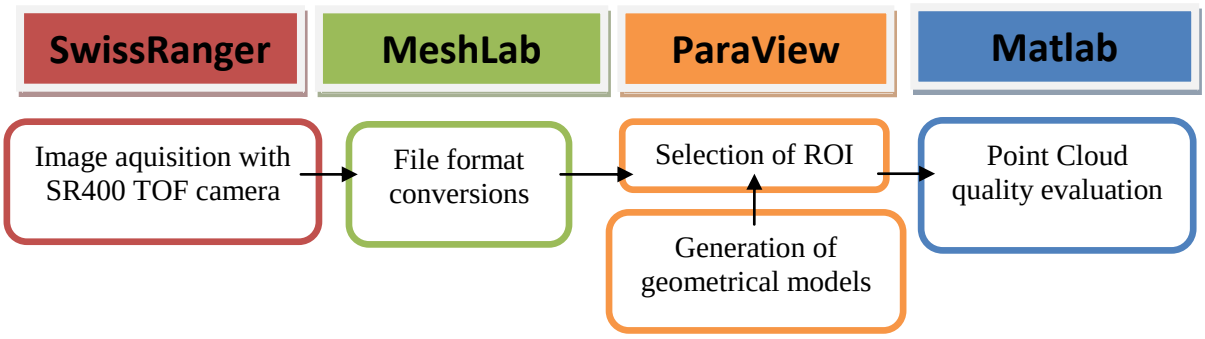
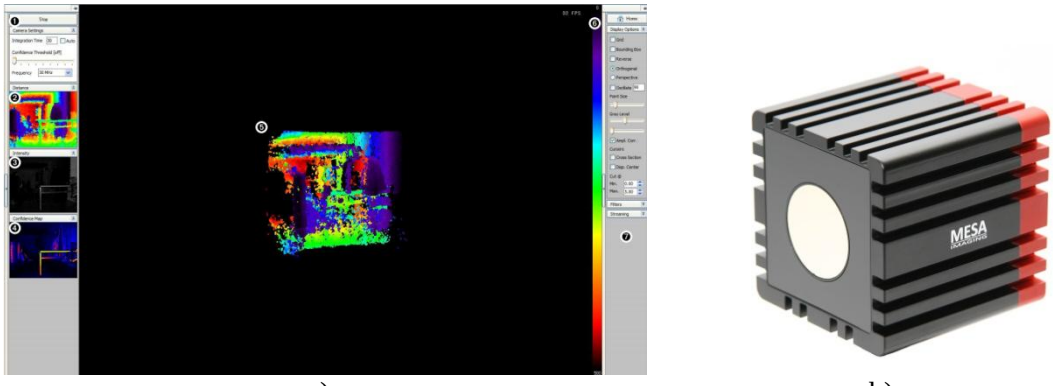


Figure 6. Overview of the whole process developed in the project. Images were first acquired in SwissRanger software by using a time-of-flight camera. Meshlab allowed the file format conversion of point clouds. ParaView was the key to select the desired regions of interest from each point cloud as well as the geometrical models we also generated with this program. Finally, the mathematical approach was fully developed in Matlab to evaluate the quality of Point Clouds.

3.1. Preprocessing

The data was acquired by a time-of-flight (TOF) camera from Mesa Imaging, model SR4000 (figure 7b'), using their software: SwissRanger. The program consists in a main window showing the depth information of the image and three secondary windows giving information on the intensity of the reflected photons (grayscale), the distance and the confidence map (figure 7a'). The software also allows applying different kinds of filters such as median or Gaussian filters. However, images were captured in a raw mode, without any manipulation. Other parameters were set in automatic mode so that SwissRanger gives the optimized values depending on the environmental conditions.



a) b)

Figure 7. Images of the two elements used to acquire point clouds: a) SwissRanger, b) SR4000 camera

Once the point clouds were obtained they went to MeshLab. This software allows changing between different file formats very fast and easily, just by loading and saving data. When

working with point clouds, different formats are used depending on the manipulation or the program.

We used ParaView to have a first view of the point clouds, evaluating their quality by visual inspection as well as determining if acquisitions were done correctly. It is also crucial to identify basic geometries such as planes or spheres in complex structures and determine the regions of interest (ROI) we will work with. ParaView allows selecting a collection of points either in the graphical interface or in spreadsheet mode (‘figure 8’). Geometrical models for validation were also produced by ParaView, a plane and a sphere.

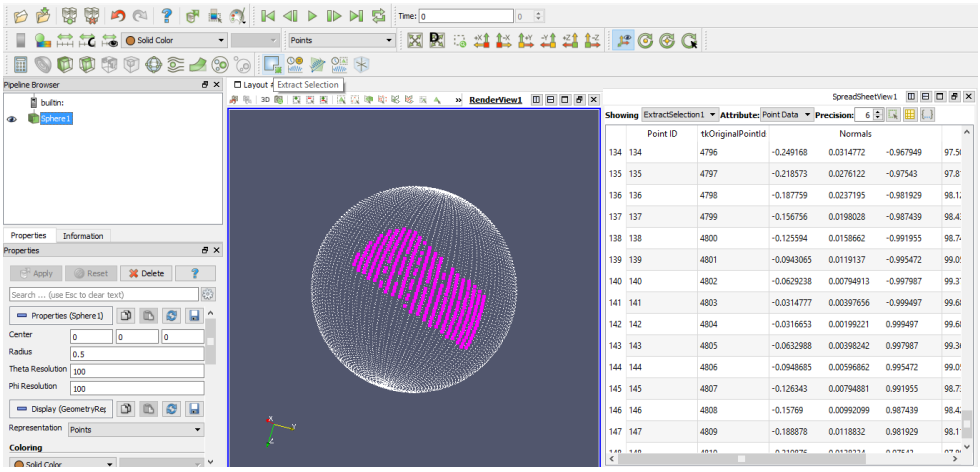


Figure 8. ParaView interface showing the selection of a ROI from the spherical model. In the right window the SpreadSheetView is shown as a numbered collection of points and its characteristic normals as well as their coordinates.

Finally, Matlab was the main tool we used in order to develop our mathematical approach. All the operations are written in scripts that act over the point clouds thanks to the Computer Vision Toolbox.

3.2. Mathematical procedure and Matlab code

To evaluate the preprocessed point cloud data we wrote two scripts capable to perform our mathematical approach. The first one, called M1, is shown in ‘figure 9’.

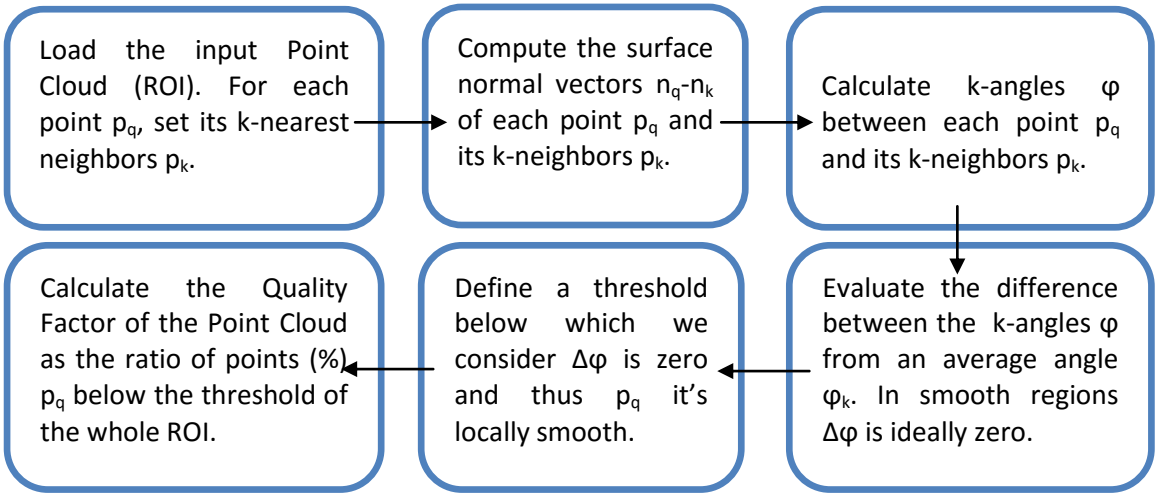


Figure 9. Schematic representation of the mathematical development in the Matlab code.

The procedure starts by writing two matrices for each input point cloud (ROI). Therefore, the process exposed in ‘figure 9’ is performed at the same time for every point of the ROI through

bigger matrices instead of point by point. The first matrix p_i corresponds to the {x,y,z} coordinates of every query point p_q, while the other matrix p_j hosts the k nearest neighbors of each p_q. Then, their surface normal vectors n_i (matrix of all n_q) and n_j (matrix of all n_k) are computed by a Matlab internal function. Since they are also written in {x,y,z} components, its sizes are equal to the former ones.

$$p_i = \begin{pmatrix} p_{11} & \cdots & p_{13} \\ \vdots & \ddots & \vdots \\ p_{N1} & \cdots & p_{N3} \end{pmatrix} (1) \qquad p_j = \begin{pmatrix} p_{1,1} & \cdots & p_{1,3} \\ \vdots & \ddots & \vdots \\ p_{N\cdot k,1} & \cdots & p_{N\cdot k,3} \end{pmatrix} (2)$$

$$n_i = \begin{pmatrix} n_{1,1} & \cdots & n_{1,3} \\ \vdots & \ddots & \vdots \\ n_{N,1} & \cdots & n_{N,3} \end{pmatrix} (3) \qquad n_j = \begin{pmatrix} n_{1,1} & \cdots & n_{1,3} \\ \vdots & \ddots & \vdots \\ n_{N\cdot k,1} & \cdots & n_{N\cdot k,3} \end{pmatrix} (4)$$

Once the vectors are set, the angle between each pair p_q-p_k is calculated with the 'atan2' function. Since the size of n_i (N-by-3) is lower than n_j (N·k-by-3), n_i is first expanded in order to be equal in size. A new matrix φ_i stores the angle for each pair p_q-p_k of the whole ROI. For every neighborhood, an average angle φ_k is calculated.

$$\varphi_i = \begin{pmatrix} \varphi_1 \\ \vdots \\ \varphi_{N\cdot k} \end{pmatrix} (5) \qquad \varphi_k = \begin{pmatrix} \varphi_{k1} \\ \vdots \\ \varphi_{kN} \end{pmatrix} (6)$$

The difference in angle for each pair p_q-p_k against φ_k is defined as $\Delta\varphi$ and averaged as $\Delta\varphi_k$. This final value is used to determine if the each p_q is locally smooth depending on a threshold we set near to zero. If $\Delta\varphi_k$ is lower than the threshold, that point is considered smooth. Otherwise, it is considered non- smooth, corresponding to edges or noisy regions.

$$\Delta\varphi_i = \|\varphi_k - \varphi_i\| = \begin{pmatrix} \varphi_1 \\ \vdots \\ \varphi_{N\cdot k} \end{pmatrix} (7) \qquad \Delta\varphi_k = \begin{pmatrix} \varphi_{k1} \\ \vdots \\ \varphi_{kN} \end{pmatrix} (8)$$

Finally, the quality of the point cloud is computed as the ratio of points below the threshold (smooth regions) of the whole point cloud (N points p_q). It is then multiplied by one hundred in order to make it a percentage.

$$Quality\ Factor\ (\%) = \frac{Number\ of\ \Delta\varphi_k < threshold}{Total\ number\ of\ points\ N} \cdot 100 \quad (9)$$

A second method, called M2, was also used. Although M1 and M2 are very similar, in M2 the neighbors are not calculated as the k-nearest neighbors (3D distance) but as the k-adjacent neighbors (2D distance). In addition, k is always set to 2. This implies that we only have two angles for each neighborhood and thus that $\Delta\varphi$ is calculated directly, without an average φ_k.

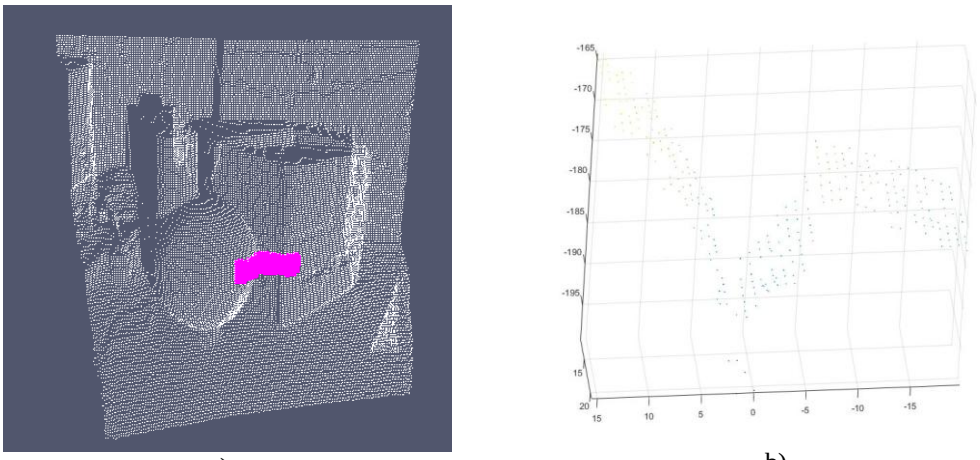
3.3. Evaluation of the Point Clouds

Point clouds were captured indoor in our laboratory, with the lights on and at room temperature. Those acquisitions consisted in random organization of geometrical structures above a table (spheres and boxes). The dimensions of the measured elements were near to the meter with mid reflectivity in order to have high-quality images. To be sure of that we used the confidence map displayed in the SwissRanger software.

In MeshLab, point clouds acquired through SwissRanger in STereoLithography (stl) file format were transformed into Polygon File format (ply) by loading and saving in different formats.
The point clouds were then loaded into ParaView, where we made the first visual inspection. If the image was well acquired we kept on working and defined the ROI. The region was defined

depending on the needs. For example, if we wanted to compare the results of our spherical model with a real sphere, our ROI was a region of the point cloud contained in the sphere. Otherwise, the ROI could be a random region of the point cloud (‘figure 10a’)

Defining a ROI was also useful in order to decrease the number of points analyzed in Matlab and thus decrease the computational time of the process. Once the ROI was selected, it was extracted into comma-separated values (csv excel file) and loaded into Matlab. There, Matlab operations were performed as explained in section 3.2 (‘figure 10b’).



a) b)

Figure 10. Acquired point cloud: a) ParaView visualization of the whole point cloud and selection of the ROI, b) visualization of the ROI in Matlab.

3.4. Validation

The validation of the mathematical development was performed by using two geometrical models: a plane and a sphere. They were produced artificially on ParaView, sectioned and processed in Matlab. In the validation, we wanted to obtain a correlation between the level of noise of the models and the calculated Quality Factor (%).

Both models were evaluated for a given range of SNR in dB for values corresponding to 100, 50, 30, 10 and 1. SNR was added through Matlab internal function “awgn”. The Gaussian noise was added in the input p_i matrix and only in the third column, corresponding to the z (depth) values of each point. Therefore, both x and y values (2d plane) remained intact and only the third coordinate was changed by the addition of white noise (figure 11’).

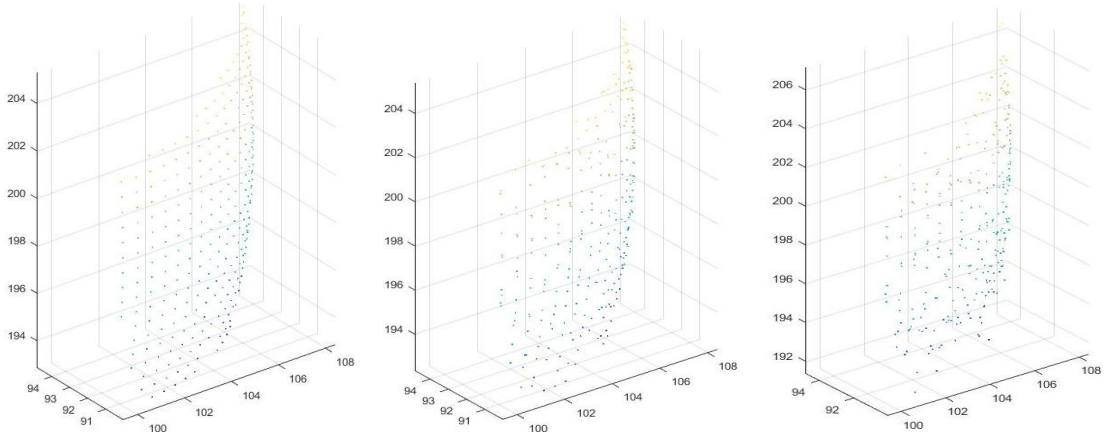


Figure 11. Effect of the addition of Gaussian noise into the ROI of the spherical model through the function “awgn”. The value of the SNR introduced distorted the imaged, allowing the evaluation of noisy point clouds to validate our procedure.

4. Results

The evaluation of the geometrical models by the first method M1 is shown in 'figure 12'. The calculations were performed in six different configurations, depending on the values used for the number of neighbors 'k' (2, 8 and 12) and the threshold 'T' (0.1, 0.2, 0.4 and 0.6). Vertical axes show the values of the obtained Quality Factor (%) as a function of the SNR (variable of the function 'awgn' added to the depth values of the point clouds, as explained in section 3.4). Each Quality Factor (%) was obtained by averaging five measures for each configuration (k,T) since they showed to have some dispersion (due to the random behavior of the added Gaussian noise). The Quality Factor represents the ratio (%) of points considered to be locally smooth for each ROI. Thus, the higher the Quality Factor the better the quality of the Point Cloud.

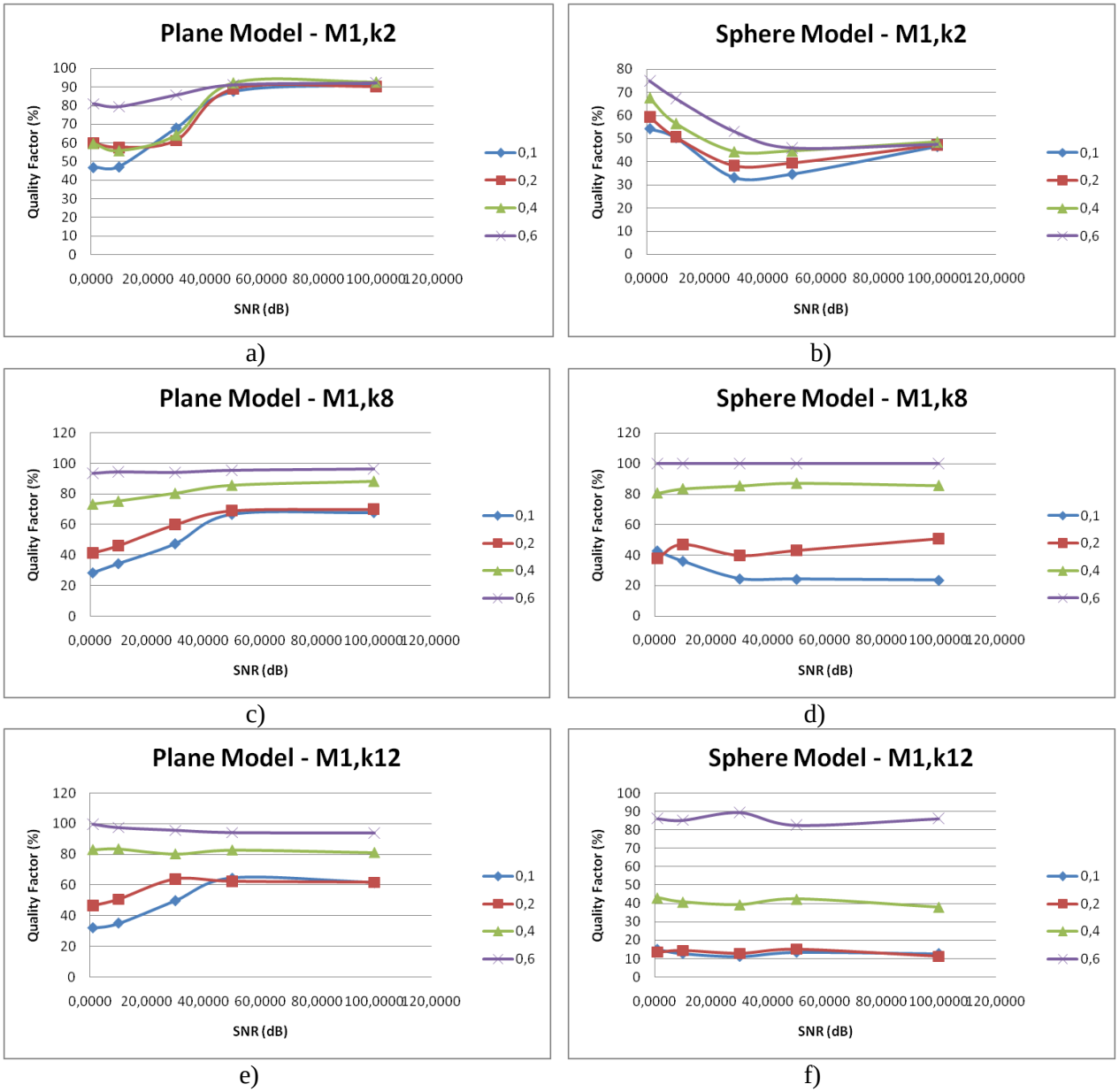


Figure 12. Results of the Quality Factor (%) as a function of the SNR for each configuration (k,T) in both the plane (left) and sphere (right) model. Values of the Quality Factor (%) for each threshold are represented in different colored lines. Both geometrical models have been evaluated for three different numbers of neighbors k=2 (a, b), k=8 (c, d) and k=12 (e, f). The planar model presents better results in general since the correlation Quality Factor (%) – SNR is usually seen. However, some points doesn't work, usually due to high levels of noise. Only when k=2 the correlation is present for both the sphere and the plane, giving rise to the hypothesis than surface normal vectors are more sensitive to noise when the number of neighbors is lower (see 'figure 4').

The Quality Factor (%) was expected to be a value between 0% (highest noise, no smooth regions in the Point Cloud) and 100% (lowest noise, Point Cloud totally smooth). Since SNR determines the amount of Gaussian noise, the Quality Factor (%) should increase as SNR does.

Moreover, thresholds represented the margin we considered for an angular difference $\Delta\varphi$ to be zero or not. Then, higher thresholds mean that more points are considered to be locally smooth and hence the Quality Factor (%) is bigger. Finally, increasing number of k was expected to give better results since it means more stability in values.

Results showed that the Quality Factor (%) as a function of the SNR works different depending on the geometry of the Point Cloud. While planar geometries always presented the expected correlation Quality Factor (%) – SNR (‘figure 12a,c,e), spherical geometries only presented the correlation when k=2 (‘figure 12b)’. When SNR was equal or near to 1, the Quality Factor lost the correlation with SNR, showing it doesn’t work for any noise level.

Furthermore, the number of neighbors ‘k’ was revealed to be a crucial factor. In general, the slope of the Quality Factor (%) function increases when ‘k’ decreases, meaning that Gaussian noise can be measured more efficiently. In the case of the sphere, values of k bigger than 2 were not able to measure the SNR since there is no correlation.

Once the first method was evaluated, the second method M2 entered the game. This method was implemented once the results of the first method M2 were obtained. Due to the bad results obtained for several SNR in planes and even worse results in spheres we incorporated many changes. Results in ‘figure 13’ show how the Quality Factor (%) increases linearly along the whole range of SNR while keeps a similar shape for each threshold for both the sphere and the plane.

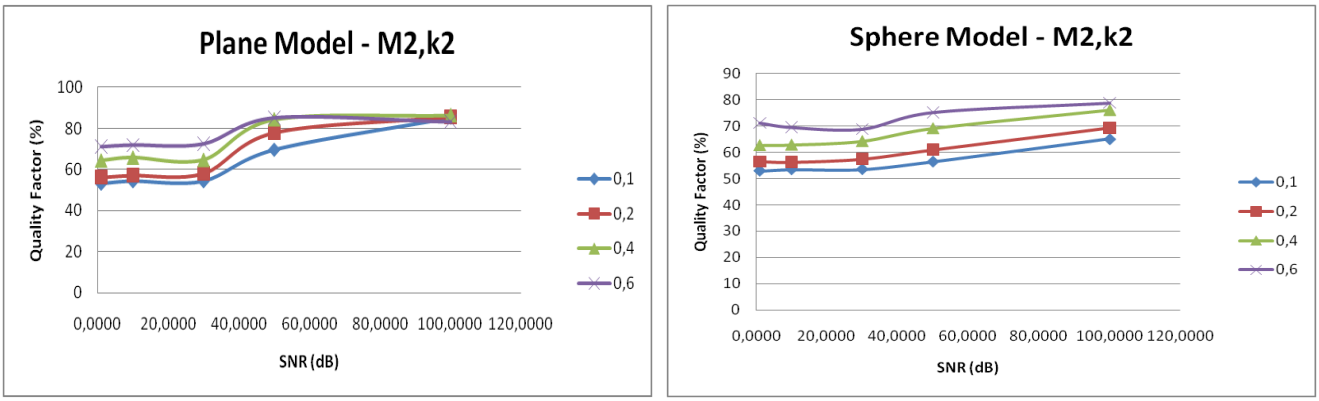
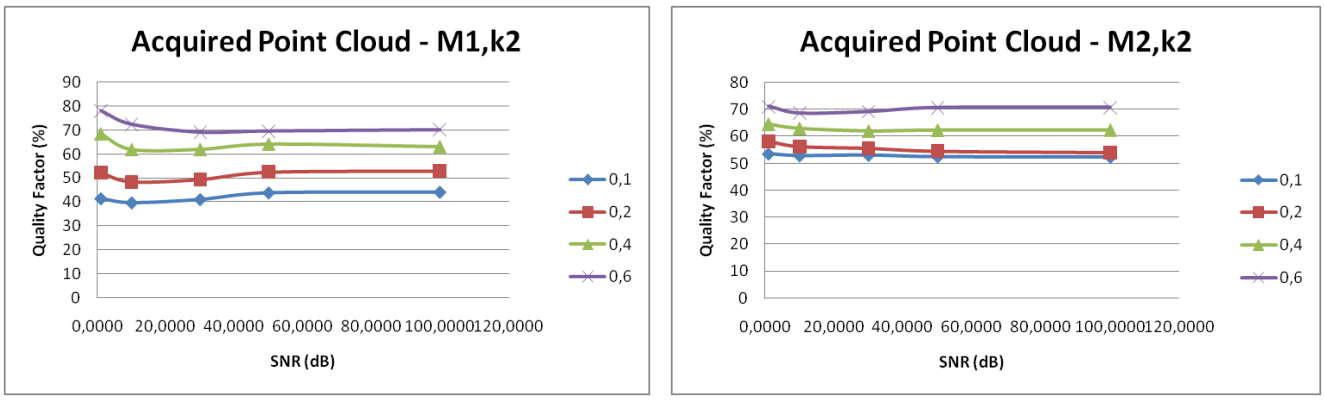


Figure 13. Quality factor (%) as a function of SNR for the second mathematical method M2 in the models of the sphere and the plane. Different thresholds are represented in different colored lines. In this case the number of neighbors k was constant.

However, when evaluating processed point clouds as explained in section 3.3, the Quality Factor showed different results (‘figure 14’).



a) b)

Figure 13. Quality factor (%) as a function of SNR and k=2 for an acquired and preprocessed point cloud: a) first method, b) second method.

In this case, the first method shows a higher slope allowing the measure of the SNR, while the second is almost flat.

5. Conclusions and future work

The large amount of values and measures of this thesis as well as the acquired deep comprehension of the point clouds provides several conclusions.

At first, it is possible to measure the quality of point clouds (i.e. 3D images) through the analysis of its normal vectors and their relative orientation. However, it is a very difficult task to obtain the best code development and the configuration of the variables. As seen, the treatment of the data can be crucial as seen in differences between the two used methods. On the other hand, differences in T and k are also crucial to work with a higher range of SNR.

Hence, future work on the point cloud evaluation requires a deepest statistical evaluation of the results since Gaussian noise behavior might be complex when acting over point clouds. More values should be taken for each configuration. Furthermore, these statistical features could give rise to other useful parameters for the evaluation such as the dispersion of values through the distribution width or similar. Statistics seemed to give different dispersion in values depending on the parameters.

In addition, different sizes of samples should be evaluated in order to determine if the number of point is also a key factor. Maybe there is a minimum value needed to make the calculations and we used quite few points.
Finally, both the parameters and the mathematical procedure developed in Matlab code shall be optimized after a deep statistical analysis.

References

[1] T.S.Huang 1996 Computer Vision: Evolution and Promise *Conference Paper.*
[2] Radu Bogdan Rusu and Steve Cousins 2011 *Conference Paper*
[3] Radu Bogdan Rusu 2009 Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments *PhD Thesis*
[4] http://pointclouds.org/documentation/