1. Open .las/.laz file (or whatever point cloud file you have) with **CloudCompare**

2. Keep the suggested numbers somewhere, you will need them after.

You need to do this because most of the time PointCloud (PCL) are registered with a geographic coordinate system, which means big numbers.

Most software are not happy about projecting number like 6725000.00 on your average Cartesian coordinate system.

X= -588500.00

Y= -6725000.00

Z= 00.00

3. At this point, we agree that you know your data and what you want to do with it.

For colors, I want to keep the classification and enhance it a bit with a P.C.V. Ambient Occlusion filter.

So, I select the classification and create RGB value from it (we will only export point coordinates and rgb value at the end).

Then, I apply the P.C.V. filter and merge it with RGB values.

4. Duplicate your nice looking point cloud.

We are going to decimate the first one, in order to fit a 2048x2048 texture. That would be exactly 4194304 points.

Before exporting it, we have to “clean it”. Meaning, deleting every ScalarFields (SF) values and clearing normals if you have computed them, because we don’t want that in the csv file and it’s a pain to edit by hand a multimillion line csv (you can do it with Vim for Cream though).

Export settings : ASC and check “Save colors as float values (0-1)”

5. Going back to original cloud, we will be using the classification to keep only the ground.

For this specific PCL, everything beside 4 in the Classification SF is ground, so we will use it as the filter value.

With the ground, we can either create a DEM to import as terrain inside UE4 (best option in my opinion but there are too many holes in this data) or create a mesh which we will use as a physical ground (since PCL don’t have any physic attached to them). The poisson reconstruction plugin is pretty good and up to date in **CloudCompare**. Still, you will have to do some cleaning afterward, in **Meshlab** for example.+-

Again, this data has really at lot of holes (water planes and castle) but is also really flat, so we will go with the lazy solution, a flat mesh created inside UE4 later.

6. Open Matlab and load the script “image\_reading.m”

filename = 'Chambord.csv'; %read csv file from CloudCompare

A = csvread(filename); %read csv file from CloudCompare

B = A(:,4:6); %New matrix B keep color information

A = A(:,1:3); %Old maxtrix A keep spatial information

A(:,1) = A(:,1)-588500.00; %Get ride of big coordinates from .las file

A(:,2) = A(:,2)-6725000.00; %Get ride of big coordinates from .las file

A(:,3) = A(:,3)-0.00; %Get ride of big coordinates from .las file

A = A - min(A(:));

A = A/max(A(:)); %Get the 0-1 scale

A = round(A, 6); %round 6 decimals

Ai = reshape(A,2048,2048,3);

Bi = reshape(B,2048,2048,3);

%image(Ai);

%image(Bi);

imwrite((Ai), 'A:\Z\_Projets\Unreal Engine\matlab scripts\spatial.png', 'png', 'BitDepth',16)

imwrite((Bi), 'A:\Z\_Projets\Unreal Engine\matlab scripts\color.png', 'png', 'BitDepth',16)

#The csv files need to be in the same folder as the script

#The only things you have to edit are the number to get rid of big coordinates and the folder where you want to save the pictures.