POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

EXECUTIVE SUMMARY OF THE THESIS

Physically Based Rendering of Animated Point Clouds

TESI MAGISTRALE IN COMPUTER SCIENCE AND ENGINEERING – INGEGNERIA INFORMATICA

AUTHOR: MATTEO POZZI

ADVISOR: MARCO GRIBAUDO

ACADEMIC YEAR: 2020-2021

1 Introduction

Rendering is a key concept in the Computer Graphics field since its very first years. Among the many techniques developed for this task, a famous one is Physically Based Rendering, born in the '80s and aiming to achieve photorealistic lighting and currently widely used in many applications such as videogames, design and many other fields involving the creation of digital images. Through the years, Computer Graphics pipelines have been optimized to work with polygonal meshes but, in recent years, the spread of scanning systems using radar or laser scanners have led to the diffusion of a different kind of 3D model using only point primitives, which have been called *point clouds*. This class of models have many advantages with respect to meshes and could be heavily used in Computer Graphics applications by substituting them. At present day most applications using point clouds use simple rendering techniques to display only the color of the model acquired during scanning, but there exist other applications involving animations which needs a point cloud to be better contextualized in the environment and a photorealistic look would be desired. In such scenarios Physically Based Rendering could be used to allow the animated point cloud having a photorealistic look under any external lighting condition. In this work we will show how Physically Based Rendering can be applied to animated point clouds as to polygonal meshes to obtain a photorealistic appearance of the model.

2 State of the Art

2.1 Point Clouds

Point clouds are a set of high-density individual points used to represent volumetric visual data, which can be computer-generated or directly captured from the real world, where each point carries various attributes like the position in the

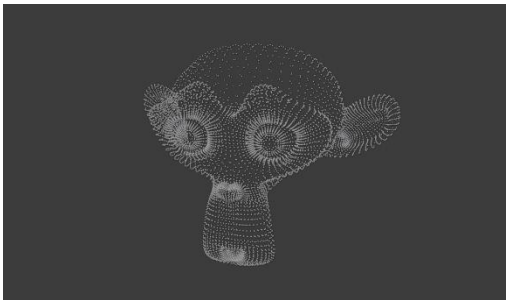

Figure 2.1 - Point cloud example representing a monkey

space and the color. Since there is no connectivity information, storage and transmission of point clouds are simpler, as points can be acquired, stored and transmitted in any order as long as the whole set of points is considered. The most common solutions used to acquire point clouds are LiDAR-based scanning and photogrammetry, but other alternatives are possible, for example videogrammetry, RGB-D cameras and stereo cameras.
The first standardization activity for Point Cloud Compression was initiated in 2014 by the MPEG group [1] and consists in two approaches: video-based, where the point cloud is projected in a 2D space and existing standards used in video encoding compression are exploited, and geometry-based, which works directly in a 3D space using a data structure called octree.

2.2 Rendering

Rendering is the process of automatically creating a 2D or 3D digital image from a scene defined by a series of objects. It is a very expensive task in terms of calculation and time and the key is to find a good balance between image quality and rendering speed. A good choice is to use algorithms that produce images with an acceptable perceived quality for the specific application we intend to use them for and don't require too much time to be computed.
Physically Based Rendering, or PBR, is an interesting and currently widely used collection of rendering techniques used in real time rendering which tries to mimic how light interacts with surfaces in a physically plausible way. An additional advantage is that it is possible to create different materials by changing physical parameters and making models look correctly under any lighting condition without the need to resort to coding hacks, as instead is needed with different pipelines.

Figure 2.2 - Environment's cubemap used in Image Based Lighting

To be physically based, the pipeline has to fulfill three conditions: satisfy the energy conservation principle, be based on the microfacet theory and use a physically based BRDF (Bidirectional Reflective Distribution Function). PBR is often used in combination with Image Based Lighting to produce even more realistic and physically accurate results, by transforming the environment into a cubemap that can be used in the lighting equations as a big light source to capture the environment's global illumination. PBR with Image Based Lighting looks a good suit to create images containing point clouds with a photorealistic look.

3 Physically Based Rendering of Animated Point Clouds

To conduct the experiments an animated point cloud will be used, where each frame is stored in a different file, to better study the impact of lighting on a non-static model. . The specific models which are going to be used as reference come from the *basketball_player* sequence provided by [2], which is a collection of 600 frames stored in .ply format.

3.1 Point Cloud normals check

The first step is to make sure that all the available point clouds come out with correctly estimated normal vectors associated to points. To make this check the software MeshLab is used, which allows to visualize the whole point cloud and the normals associated to its points.
It turns out that many frames of the animation are decorated with correct normal vectors but, in some frames, these have been erroneously estimated and are oriented inward the model (Figure 3.2). Since even the models with correct estimated normals

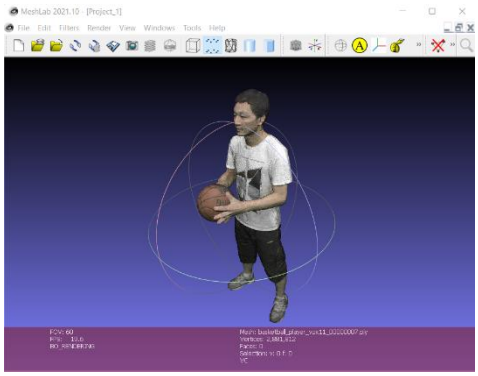

Figure 3.1 - Point Cloud frame visualized in MeshLab

look a bit rough and "squared", all frames have been subjected to normal vectors re-computation, to make the models look as smooth as possible.
The operation is performed with a filter called "compute normals for point sets" available in MeshLab, which computes the normal vector of each point by estimating the plane tangent to the surface from the point neighborhood.
After the operation, all point clouds have correct normal vector information associated to points and can be imported into Unity.

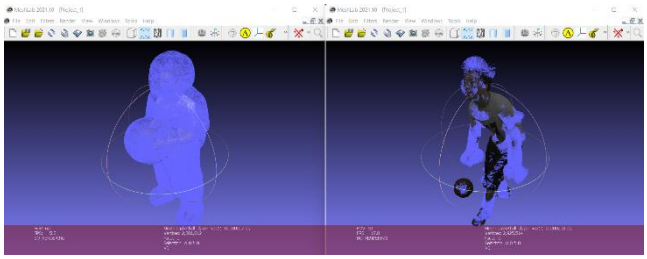

Figure 3.2 - Frames with correct (left) and wrong (right) estimated normals

3.2 Importing models into Unity and PBR shader

Unity has no native package allowing to import and directly use in a project a .ply asset, so it is necessary to use a custom importer to load the point clouds into the project. For this purpose, a package available on GitHub called Pcx [3] developed by the user Keijiro has been added to the project and used to import the models. The package also provides a basic unlit shader that displays the point cloud with its original points color, which will be used, with a minor modification, as baseline to compare the PBR with. However, the importer has an issue: it hasn't be designed to import point clouds with normal vectors and hence normals are not parsed and included in the imported models. So, the parser needed to be slightly modified as well to include normal vectors.
After having solved the issue, the PBR shader can be developed. To be faster in the development, the Unity's Universal Render Pipeline with ShaderGraph support is enabled. A lit shader graph, which contains all the parameters needed by PBR like metalness and smoothness, is created and applied to a frame of the point cloud. The comparison between the point clouds rendered

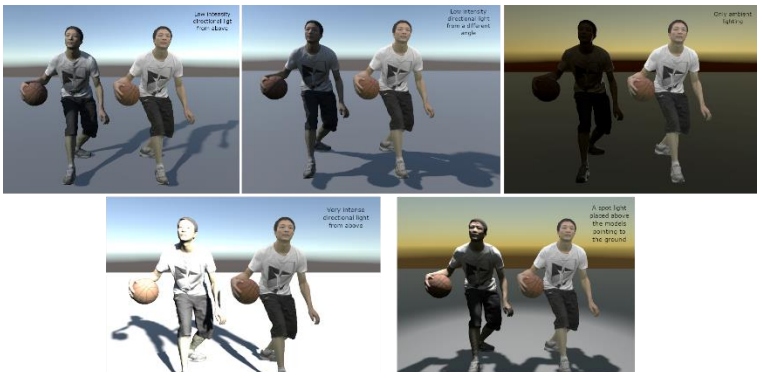

Figure 3.3 - Comparison between the point clouds rendered with PBR (left) and unlit (right) shaders

with the two techniques and with different lighting conditions is shown in Figure 3.3.
As can be noted, the model rendered with PBR looks different with respect to the incoming light direction and intensity, as opposed to the unlit one which instead looks exactly the same under any lighting condition.
In the next sections a couple of relevant scenes developed during the work are exposed.

3.3 Test scene: background 360° video of an indoor basketball court

In this first test scene the objective is to try to better contextualize the point cloud model by inserting it into an environment it might be found in. The background choice fell on a 360 degrees video uploaded on YouTube by the Columbia International University representing an indoor basketball court with some people playing around. The video comes up in the equirectangular format, so it can be easily rendered on the scene's skybox using a Render Texture and the built-in Skybox/Panoramic shader.

Figure 3.4 - Depth issue: when zooming the camera, the background doesn't zoom with the point cloud

A few issues in the rendered scene can be spotted related to the integration of the point cloud with the video: since the background is rendered on the skybox, there is no sense of depth when the camera is moved towards the model, and when the camera is rotated the model looks like fluctuating over the background.
In this setting nothing can be done to solve the other issues since the Unity's skybox represents the content of the scene placed on an infinite distance from the camera and will always be rendered behind any other 3D model inserted in the scene. To produce realistic scenes, other 3D models need to be placed together with the point cloud, with the background having a content on a sufficient distance to prevent unrealistic behaviors when the images overlap.

3.4 Test scene: outdoor basketball court in a daylight environment

Since a background video alone is unusable due to the introduced issues, other 3D models are needed to be placed in fore-midground to contextualize the point cloud model. Regarding the 3D model, the choice went to an outdoor basketball court which is shown in Figure 3.5, while for the background a 360 degrees video recorded on a sunny beach in California with some beach volley fields has been chosen. After having adjusted some light intensity parameters to better simulate daylight, the point clouds are inserted into the scene.

Figure 3.5 - 3D model of an outdoor basketball court

Although the unlit point cloud, being very bright, suits quite well the surrounding environment, it still is not as realistic as the point cloud rendered with PBR which shows a darker appearance due to occlusions and self-occlusions of the model, noticeable on the neck of the player, on the t-shirt and on the lower part of the ball. When using PBR many smaller details become more distinguishable, like the t-shirt folds which due to the harder shadows produced by self-occlusion become more highlighted, for example regarding the upper part of the clothing.

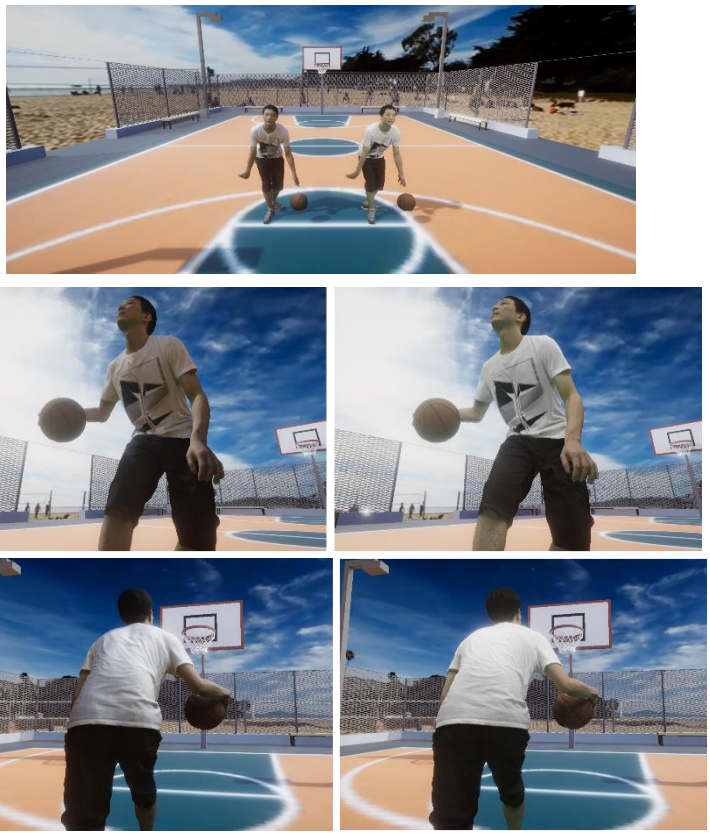

Figure 3.6 - Comparison between PBR (left) and unlit (right) shaded point clouds in daylight environment

It is also clear how, with PBR, the incoming light direction is visible on the model, being the left side of the player in shadow, while this is not happening on the unlit point cloud. In conclusion it can be said that in a daylight environment the PBR point cloud is more suited to realistically reflect the external lighting condition with respect to a point cloud using a standard shader.

3.5 Test scene: playing with metalness and smoothness parameters

One of the most powerful features provided by PBR is its capability of emulating metallic and glass-like materials, thanks to the microfacet theory and the energy-conserving principle. In this scene the focus will not be on integrating a point cloud inside an environment but on demonstrating how point clouds, as meshes, can be used to render different type of materials, for example perfect

mirrors. As background, various environment textures rendered on the skybox will be used without the basketball court model. To render a material representing a perfect mirror, the smoothness and metalness parameters are both modified to 1.0. In Figure 3.7 the point cloud reflecting the environment as a perfect mirror is shown for a couple of sample environments.

Figure 3.7 - Point cloud rendered as a perfect mirror placed in two sample environments -

It is evident how the point cloud, when placed in different environments showing different lighting dynamics, still looks physically accurate regardless of the specific environment.

4 Applications

Although at present day the use of point clouds might still not be very affordable in many circumstances due to the high cost of producing them, there are many fields that could potentially benefit from a wide exploitation of such models. Three example macro areas are:

Learning: point clouds could be used in guided tours, by acquiring guides as animated point clouds with audio that can be rendered in Augmented Reality to give tourists higher flexibility on the tour, or in sports where specific movements need to be learnt, like martial arts and dance. The use of a point cloud representing a trainer of the discipline can be used to effectively study the movements to reproduce the target technique.

Advertising: point clouds can be used to sponsor clothing online, in place of the many pictures usually needed to show the item from different angulations or also generate an animated avatar wearing the item itself and reflecting the customer movements showing how the item fits.

Entertainment: point clouds could be used in videogames to render specific objects and save the time needed to create a 3D mesh by designers, or in animated movies and immersive experiences.

The use of PBR is not mandatory in any application, but as shown, using point clouds with just their own color results in having a model disconnected from the environment in which is placed and gives a strange look to the scene. By using PBR, the point cloud fits better into the scene and gives a better overall appearance to the rendered image.

5 Conclusion

In this work the goal was to demonstrate how a point cloud rendered with PBR could achieve better realism during visualization with respect to a point cloud rendered with a basic unlit shader using just the original vertices colors of the model. It has been observed how with PBR a point cloud better integrates with the surrounding environment in any lighting condition, by casting shadows due to direct lighting and self-occlusions. It was also shown that it is possible, for a point cloud using PBR, to render different kind of materials as glass-like and mirrors in a physically plausible way. As point clouds are becoming more and more popular, using PBR to render them would be a benefit for many applications, going from simple model visualization to immersive experiences in Virtual or Mixed Reality.
This work was intended to show the potential of using PBR to render point clouds but there are many areas and topics that were left uncovered and beyond its purpose, which can be subject of future developments for improvement, like the efficiency of the process to visualize an animated point cloud.

6 Bibliography

[1] MPEG-PCC, "Introduction to the MPEG-PCC project," 2019. [Online]. Available: https://mpeg-pcc.org/.

[2] Y. Xu, Y. Lu and Z. Wen, "Owlii Dynamic human mesh sequence dataset," in *ISO/IEC/JTC1/SC29/WG11 m41658, 120th MPEG Meeting*, Macau, 2017.

[3] Keijiro, "Pcx," GitHub Repository, 2021. [Online]. Available: https://github.com/keijiro/Pcx.