Computer Graphics Project
Shell-based texturing

Authors

Naglis Naslenas - s223312

December 17, 2023

Contents

1 Introduction

Shell texturing in computer graphics involves applying numerous textures to the surface of a 3D model to obtain the look of different materials such as short grass, fur or hair. The aim of the technique is to replicate real-world surfaces by adding depth and complexity to objects without the need for complex geometry that otherwise would be expensive to render (especially in real time). Due to that, shell texturing is a popular technique in computer games.

This technique works by applying a texture to a surface, and applying subsequent textures to the same surface, but a bit offset and potentially sligtly modified from the previous one depending on the desired effect. For this report, the focus will be on the implementation of shell texturing for grass-like surfaces, but a similar approach can be used for other materials.

2 Creating Texture

The grass texture used in the project is procedurally generated using basic random noise. While it might be more realistic to use a a different noise function, such as Perlin noise, the simplicity of the random noise function made it easier to focus on the implementation of the shell texturing technique. Used function generateRandomNoise() creates an array of texture data of a specified width and height, with values between 0 and 255, thus creating a black and white noise that is used as a basis for the grass texture.

This is then used as a luminance texture where white/whiter parts indicate protruding grass blades, and the dark parts will remain flat and transparent int subsequent layers.

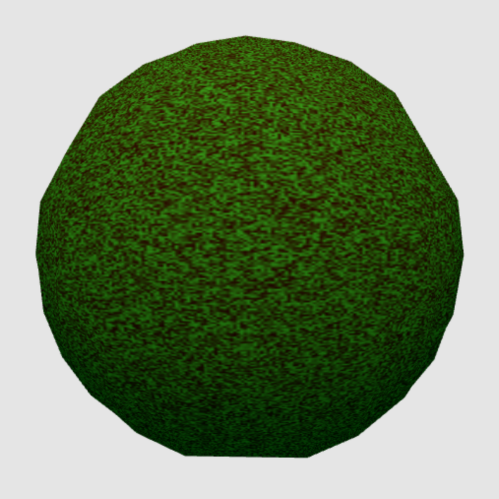

Figure 1: Initial Noise texture, where luminance is represented in green

3 Multiple layer setup

For any subsequent (besides the first) layers it was set up that another sphere with a fixed increase in radius is rendered and for fragments in that sphere that have luminance values below a certain number, those fragments are removed (alpha 0.0) and with appropriate alpha blending, those fragments are then transparent. As for the remaining fragments above the threshold, they are once again displayed.

The threshold value is increased with each subsequent layer of of the texture, so that the imitated grass blades are not of a flat nature, but instead are pointy towards the top.

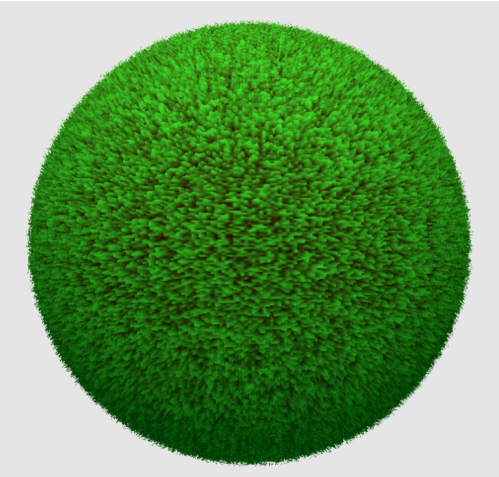

Figure 2: Sphere with multiple texture layers

As seen in figure 2, the sphere already looks quite convincing even without any additional lighting, but seemingly only from the varying luminance data based off of the texture, but there is room for improvement.

4 lighting

For more realistic lighting, grass blades should be more dramatically varying - the tops of the grass blades should be significantly lighter as opposed to the grass blades deep within, therefore a layer alpha value was introduced equal to *currentLayer/totalNumberOfLayers*. This gave an even more desired look to the object, closely resembling actual grass. To see how further it interacts with the light, a directional light was tuned for the grass to obtain both green and a bit of red color to simulate how it would look in a warm (sun-like) environment.

5 User interfaces

To "play around" with the grass texture, two sliders where introduced to control how many shell texture layers should the sphere have as well as another slider to control the radius

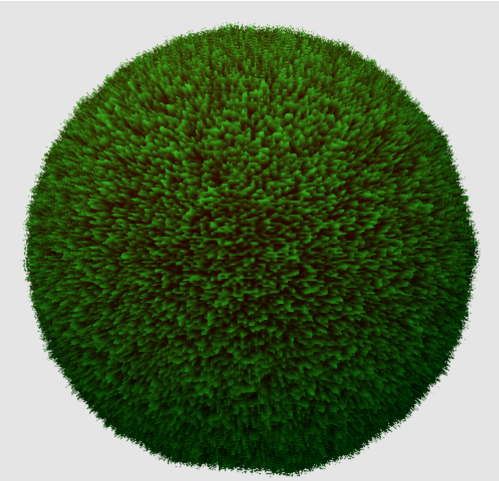

Figure 3: Layer-based alpha value effect

increase between the different layers, so the code also had to be adjusted to accommodate to the potential of the change of this parameter while running, and also, that sensible values of these two parameters would not break the illusion of a highly detailed grass.

6 Conclusions

While the effect of shell texturing is surely impressive when looking at correct angles, it certainly has its limitations when looking at it at not so optimal perspectives. In figure 4, at the top of the sphere it is noticeable where the subsequent layer lies, creating a distorted, blurry image.

Assuming that the limitation of shell texturing are taken into account, I still see an enormous use for this technique to have numerous high quality materials rendered on screen simultaneously.

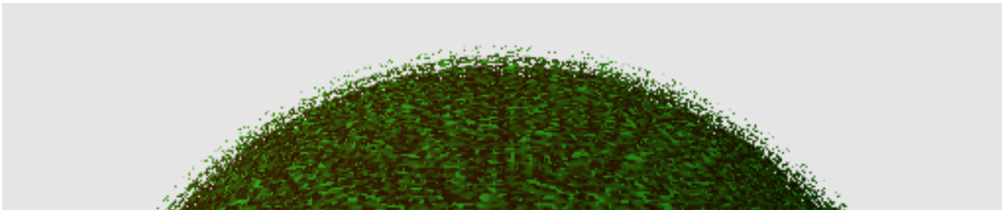

Figure 4: Visible texture offset