Principles of Rendering: Realtime Rendering Project

Alex Cowell
National Centre for Computer Animation

Abstract

This document aims to explain the process of creating the final render of my chosen object, a pencil, in OpenGL, from the analysis of the shading properties of the object to the end results of my shader implementation, and a comparison between the original reference object and final results.

1 Introduction

The pencil I picked to recreate, as seen in Figure 1, has several distinct qualities that can be broken down into layers that I recreated using shader techniques. I broke these down step by step and worked out how to implement these into a shader program.

Figure 1: *My reference object.*

The pencil can be broken down into four distinct areas which I further analysed to create shader effects for- the body of the pencil, the top which is painted black, the sharpened area and the graphite at the end. The body of the pencil is covered in a varnish that creates a wide specular highlight with fuzzy, poorly defined edges. Looking closely, as can be seen in Figure 2 there are many tiny marks on the surface like scratches from the wood grain, that make the surface appear almost blocky. The 'blockiness' seems to be due to the sheer amount of tiny grain lines marking the surface, which, since they are all from one end of the pencil to the other, make the gaps in between appear to be tiny rectangular blocks of colour. The body also has areas of dirt that darken the colour slightly, in large but infrequent patches.

Figure 2: *The body of the pencil.*

The area of the pencil that has been sharpened also shows the tiny marks on the surface which vary the colour slightly, and is darkened greatly by dirt that has collected, moreso than the body (evident in Figure 3). Without the varnish coating of the body, this area is matte and has no visible specular highlights in the lighting that I replicated.

The graphite on the end of the pencil has a slightly more well defined highlight than the body of the pencil, which reveals angular faces on the material as well as darker lines, similar to the wood grain but which stretch from one end of the graphite area to the other without breaks, all of which can be observed in 3.

Figure 3: *The end of the pencil, showing the graphite and the sharpened sections.*

The top of the pencil is a dark black in colour, coated in the same varnish that the rest of the body is. No patterns are visible on this surface unless light is shining directly onto it, wherein it is revealed that the same marks that frequent the body of the pencil still show through, although are no longer a variation in colour but just affect the specular highlight, creating dark lines on the highlight, shown in Figure 4.

Figure 4: *The top of the pencil.*

Wood patterns have been replicated by shader programs often, which use noise to replicate the natural look of the wavy grain. While my pencil does not have the kind of prominent wavy pattern that first comes to mind when thinking of wood, this was my starting point, and I looked into how wood shaders had been constructed previously, coming across The Book of Shaders [Patricio Gonzalez Vivo and Jen Lowe 2015] who explain in detail how many effects can be created with GLSL shaders, including wood, using multiple different noise functions.

2 Method Overview

There were many steps involved in creating the shader program to represent the pencil. After modeling the basic shape and importing the models into my program as obj files, I started by working on the fragment shader that would apply the complex patterns of texture to the object.

The most prominent feature of the pencil, the wood pattern, was achieved by several different functions inside the fragment shader, which used the Phong model to calculate lighting. As I mentioned in the introduction, a key ingredient in many wood shaders is noise.

There are many different ways to generate noise, but the one I settled on was 'value noise', which I derived from a detailed implementation from Shadertoy [Inigo Quilez 2013].Generating value noise is simple- for a grid of vertices, each is assigned a random number and whenever a value is needed for any point, the four values of all the vertices of the grid cell are bilinearly interpolated to produce the value [Inigo Quilez 2013]. An example of a 2D texture generated using value noise can be seen at Figure 5.

Compared to other methods such as Perlin noise, which involves assigning gradients to each point, working out the dot product, and then interpolating from there [Perlin 1985], value noise requires less calculations and would therefore be quicker to compute. Perlin noise, on the other hand, produces 'higher quality noise', as the method it uses results in it being less likely to have large chunks of similar values. As this is a real-time rendering project, speed is important, however it's unlikely that the difference would make an impact on modern machines on the scale of a pencil. That being said, I stuck with value noise as it was painless to implement and produced results that I was satisfied with.

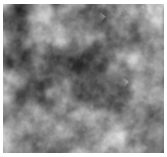


Figure 5: *Value noise taken from [Inigo Quilez 2013].*

Aside from the shaders themselves, I also applied camera effects to the scene to increase the realism of the overall render, most prominent of which is the implementation of shadow mapping to make the pencil both cast a shadow and receive shadow information to shade with. I achieved shadow mapping by first rendering out a texture to a framebuffer object from the view of a light, which stored depth values from the objects it was pointing at. This is then passed onto the shaders, which use the data and work out whether points on an object should be rendered as in shadow or in light from the depth values and the position from the viewpoint of the camera. I also added an environment map to the scene to increase the realism of the reflections on the pencil. This process takes a cube map, six images that represent the sides of a cube, usually photographs, and bases the reflection of objects rendered on them, as if the object was getting light from the scene.

3 Results

The first effect creates lighter and darker lines which are short and thin, which is a feature that is prominent on the pencil itself. This was achieved with fractal noise generated from value noise. I used this resulting value to mix two colours that I chose to get a result close to the pencil, and it created the line pattern, as can be seen in Figure 6.

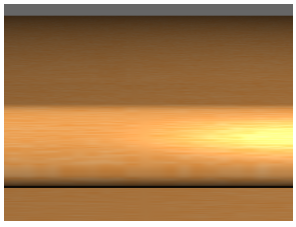
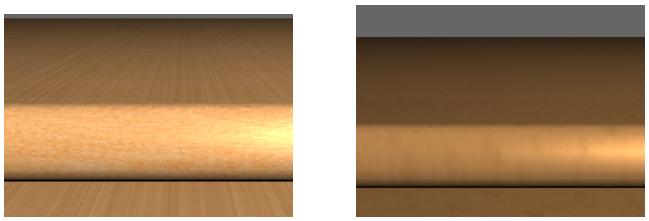


Figure 6: *The lines created by the value noise.*

The second effect came as an accident, and resulted from playing around with another function that created fractal noise (from The Book of Shaders [Patricio Gonzalez Vivo and Jen Lowe 2015]) and increasing its scale until the individual cells were easily visible, seen in Figure 7 (a) applied on top of the previous effect. This created an effect when merged with the lines, however, that closely matched up with the reference object and its blocky look.

Finally, using noise once more, I apply a layer that adds a dirt effect, by darkening patches on the pencil. This can be seen in Figure 7 (b) without any other layer of the shader. All these are mixed together, along with values from the shadow map and the environment map to create the final shader seen in Figure 8



(a) *Blocks of noise* **(b)** *Dirt noise*

Figure 7: *Two more shader layers, (a) with fractal noise to create the block effect and (b) with noise to create a dirt patch effect.*

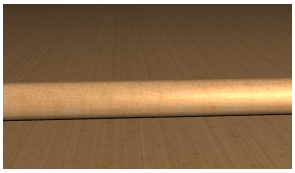
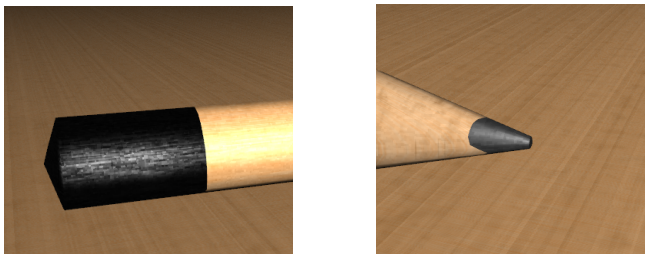


Figure 8: *Final result of the shaders applied to the body of the pencil.*



(a) *Top of the pencil* **(b)** *End of the pencil*

Figure 9: *The other areas of the pencil, each also used a multitude of different noise functions to achieve a likeness to the original reference model.*

I'm happy with the state that I managed to get my final render in, and the likeness that it shares with the reference. However, there are many areas where I feel I fell short and would improve or add on, such as more complex lighting (e.g. ambient occlusion), camera effects such as depth of field which I failed to touch on, or a method to increase the quality and realism of the shadows. It may well also be worth exploring more complex noise to achieve something closer to the pattern on the original pencil, as although my method is competent enough at replicating the look, if zoomed in on the inaccuracies start to become more apparent.

References

INIGO QUILEZ, 2013. Value noise (2d). https://www.shadertoy.com/view/lsf3WH.

PATRICIO GONZALEZ VIVO AND JEN LOWE, 2015. The book of shaders. https://thebookofshaders.com/.

PERLIN, K. 1985. An image synthesizer.