Edinburgh Napier UNIVERSITY

Coursework Report

Timothy S. Sanderson
40219124@live.napier.ac.uk
Edinburgh Napier University - Computer Graphics (SET08116)

Abstract

This is a project which demonstrates the implementation of basic lighting, shadowing, and texturing effects, in a 3D graphical scene.

Keywords – Graphics, Hierarchy, Phong, Shadows, OpenGL

1 Introduction

This project demonstrates basic, graphical concepts through the use of a 3D environment, rendered using OpenGL. In this scene the main focus is a structure made of numerous moving spheres. The spheres represent a hierarchical relation, as well as providing a good demonstration of the reactivity of the implemented lighting effects. Due to their rapid movement it is easy to see how efficiently the highlights are tracking across their surfaces, and how the shadows are being cast behind them.

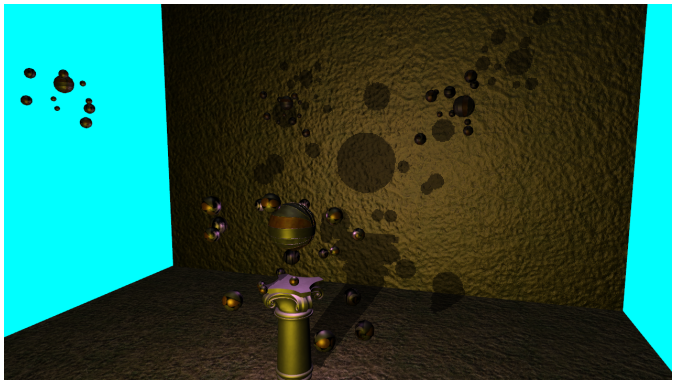

Figure 1: **Old scene overview** - A good showcase for basic techniques in the scene

2 Implementation

The following are the concepts implemented in the scene to give its current appearance.

2.1 Textures

All objects in this scene are textured. By applying a texture to an object it can be given aesthetic detail that makes it more pleasing to look at, and easier to discern from other objects.

A more complicated use of textures is the normal map. A normal map is used by a shader to manipulate how light shines on an otherwise flat surface. In Figure 1 the back wall is a completely flat plane, but thanks to normal mapping it looks like a roughly textured wall.

The final version of the project uses a floor created with a height map. A height map is a texture used to create geometry based on colour values. The closer to white the texture is, the higher the y-position of the corresponding vertex. Although much more dramatic terrain can be generated from a height map, simply creating a gentle hill is beneficial to this scene for two reasons. The first reason, it elevates the centre of the scene, bringing focus onto the important elements of the environment. Secondly and more simply, it prevents the ground from being a large flat plane, which is a visually unappealing thing to look at.

Textures are an incredibly important thing when it comes to post-processing. When a scene is rendered it outputs to a frame buffer. By default a scene is rendered to the screen's frame buffer. However, the output can be changed to a frame buffer object, and from here the rendered scene can be saved as a texture. Once a render has become a texture, that texture can be used for post-processing

2.2 Lighting

Objects within this scene implement Phong illumination. This lighting model implements the use of ambient, diffuse, and specular lighting.

On the right of Figure 2 can be seen the ambient light. It's set to be very low, but some details can still be seen. This lighting type is most simple of all. Ambient light is applied uniformly, and without regard to light or view direction.

The areas where details can easily be seen are the ones demonstrating diffuse lighting. This is the light which is directly hitting, and illuminating a vertex. Diffuse lighting changes its position on an object based on the direction of the light source. As such a vertex on a spinning object will not always be illuminated, and the maths needs to account for this.

Bright yellow highlights on the left are an example of a specular reflection from the yellow light source. Specular highlights are special because their location changes based on viewing angle, the position of the vertex, and

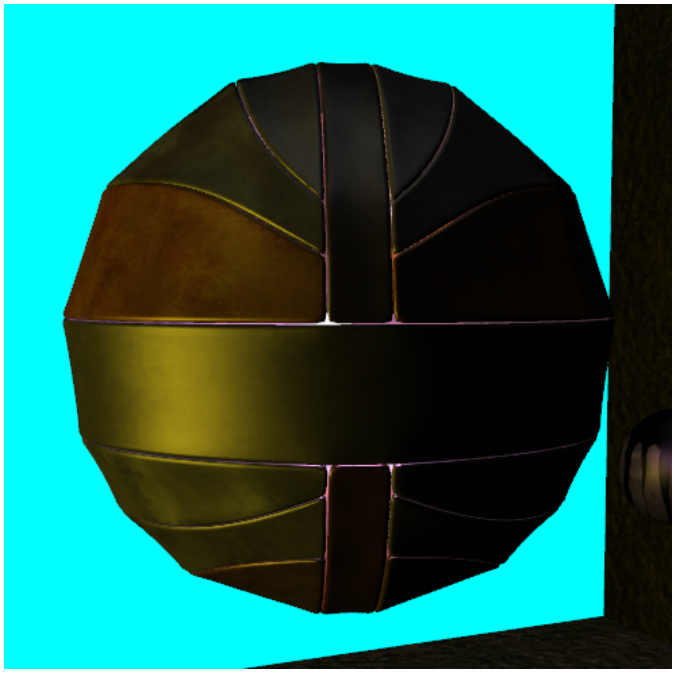

Figure 2: **Lighting** - Lighting effects demonstrated on an orb

the location of the light.

To test the relation of a vertex to light, the vertex's normal is used. When combined with the direction to the light in a dot product this gives information about the angle between the two and, from there, the degree of lighting can be calculated.

2.3 Shadows

Shadows are the relatively complicated and mathematically intense technique. To know if something is in shadow it needs to be known if there is an object between it and the light. In order to find this out the scene first has to be viewed from the position of the light. This data is saved to a texture which allows the red value to be used to determine how far away that vertex was from the camera. Due to the nature of colours holding whole values to up to 255, this means there is a very limited capacity for detail. In this scene the shadows are calculated from objects up to 300 arbitrary units away, so that shadows are far reaching, but do not lose too much precision. Then, when rendering the scene to have shadows implemented, the vertex needs to be operated on with matrices from both the camera's view and the light's view, in order to see whether it lies within the cast of a shadow.

3 Future Work

A major drawback of this scene is the lack of a skybox. The implementation of a skybox instantly gives the scene a vastly increased aesthetic appeal. In both Figure 1 and 2 there can be seen a bright blue colour in the background. This is a default appearance and not one that looks nice. Hence, a skybox being the next thing that will be added.

The scene could also do with a more cohesive setting/theme, whereas it currently is visually like a tech-demo. Although functional, there is much that could be improved to give the scene more personality while still demonstrating the same concepts. Not only should the scene be more interesting, but it should also be larger. In the current iteration the lights all fight to be seen over the others, and although this does demonstrate the ability to use multiple lights, it makes the scene feel cramped.

4 Conclusion

Overall, this scene has met the requirements made for it. Although it might lack the personality that would make it more interesting to look at, it is still an effective exercise in graphical effect processing.