CITS3003 Graphics Project Report

Kane Alexander (22710428) and Isaac Bergl (22710992).

1. Overview

This project was a success as we produced a working version of the project with the desirable features as well as numerous small quality of life improvements that could improve the user experience. The final product was not achieved without complications; namely the theoretical and then practical implementation of both the directional and spotlight slowed the project's initial completion.

2. Specific Features

2.1 Camera Rotate X,Y,Z and Texture Scale (task a,b)

Camera view rotation was implemented through updating the view model matrix. Changing the location, scale and rotation of the object was more complicated as to make it per the instructional videos. The correct order of rotations was found through research of 'general rotations' which duplicates the video examples[1].

2.2 Ambient, Diffuse and Specular Lighting (task c)

When applying the different amounts of lighting we needed to choose an appropriate lighting scale. Ambient, Diffuse and Specular all had a constant scale of 1, whilst shine needed a constant scaling factor of 50 to produced acceptable results.

2.3 Clipping and Reshaping (task d,e)

To improve object clipping the value of 'neardist' was reduced to a value of 0.001 after trial-and-error correction. To correct the reshaping, we added a conditional to when the width was less than the height to dictate the correct scaling.

2.4 Light Reducing with Distance (task f)

In order to ensure the light reduces properly with distance we added light attenuation modelled by the standard quadratic equation $1/(a + bd + cd^2)$ with a,b,c being different parameters for each light (we wanted less light decay for certain types of lights)[2].

2.5 Lighting in Fragment Shader (task g)

In the fragment shader, uniform variables previously passed to the vertex shader were updated to be passed to the fragment shader where Blinn-Phong shading calculations were performed. Varying variables such as vertex positions and the normals were calculated the vertex shader and passed into the fragment shader. Other variables such as the brightness and RGB of the lights had to be created as uniform variables.

2.6 Specular Highlights (task h)

To ensure the specular highlights always tended to white instead of the RGB of the light or the object we averaged all components of the specular light and applied them uniformly. This was particularly hard to test in a practical environment, so we created a control setting and included photos below. In the first image the normal specular product is applied, and the object does not tend to white, whilst in the second visible white highlights are prominent.

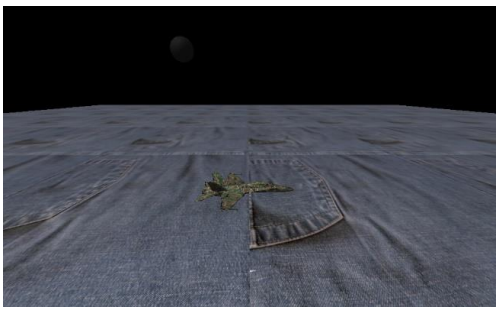
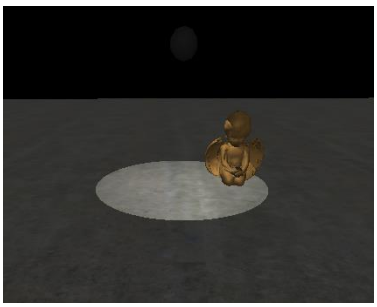
2.7 Directional and Spotlight (task i,j)

The addition of a directional light source was one of the more difficult tasks to implement as the light can only be dependent on the light source's location from the origin. We had to ensure complete separation from not only the cameras view but also the objects themselves to create a truly directional light. This was achieved by finding the origin with respect to the view model and taking the vector from the origin to the light source as the incoming light for every fragment. Using these vectors in the Blinn-Phong shading model created a directional light.

The spotlight was implemented similarly to a point light, with the exception that any light outside of a set radius was diminished. This was constructed through the dot product of the normalized vector that the lights pointing in, with the normalized vector between the light to the point. This dot product returns the cosine of angle between the vectors, and there follows the arccosine of this value will return the spotlight angle. Values returned higher than a variable angle are not 'lit up' by the light whilst those smaller are.

Additionally, we implemented an attenuation for the spotlight which was calculated as; the cosine of the angle to the spotlight direction vector raised to some power. The larger the power the sharper the intensity drop off from the centre of the spotlight.

The ability to dynamically change the RGB of all newly implemented lights was also preserved with the creation of these new lights. When position/scale of the spotlight is selected, left click and middle mouse allow its illumination direction to be changed interactively.

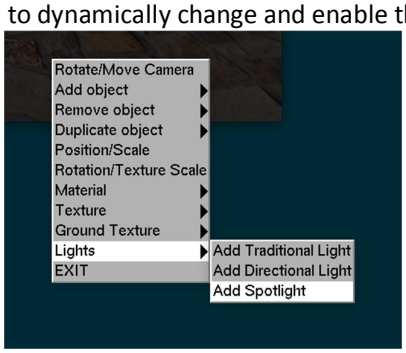



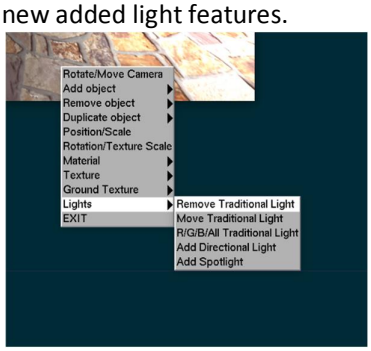

2.8 Object Deletion and Duplication (task j)

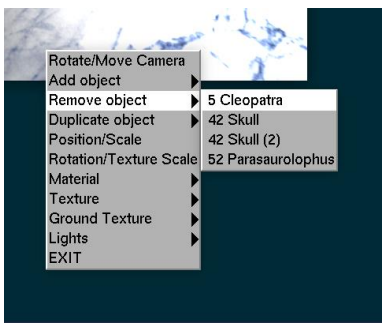
The chosen deletion and duplication implementation is consistent with the initial object creation menu existing in the project template. Two menus of dynamic size are updated with each addition or removal of a scene object. These two menus allow the user to pick what object to duplicate or remove; duplication copies the texture, scaling, meshId and rotation of the chosen object while removing deletes all the object data. Deletion removes the object completely and therefore, allows more objects to be added until object max limit is reached. Duplication does not copy the location of the object chosen, as this seems unintuitive to the user as the duplicated object would be immediately 'lost' inside the original object.

If two or more objects are the same in a scene, the menus provide each object with a unique number identifier according to the order in which they were added. This is a quality-of-life improvement for the user so they can keep track of each object in the menus.

Since the lights implemented in part i/j are technically scene objects, it is fitting to add them to a menu so they can be deleted and added at the user's desire. This required a rework of the given light menu to allow it to dynamically change and enable the new added light features.







3 Bibliography

1. https://www.brainm.com/software/pubs/math/Rotation_matrix.pdf
2. https://lms.uwa.edu.au/webapps/blackboard/execute/content/file?cmd=view&content_id=_2245852_1&course_id=_56252_1