Assignment 4 - Normal Mapping and Reflections

Due Sunday by 11:59pm **Points** 100 **Submitting** a file upload

Assignment 4 – Normal Mapping and Reflections

This exercise will help you put in place the understanding about normal mapping and reflections.

I recommend reading the exercise description and guidelines as early as possible and starting early as the exercise is challenging and requires team work.

General Guidelines:

1. Submission should be made electronically as before.
2. Any theory should be added in Word/PDF and sketches can be added as jpg attached to the document
3. Submission of this exercise is **in couples as set**
4. Partners should implement the project together with full knowledge of the entire code.

Submission Guidelines:

1. The project should be submitted as a zip file with the exercise and the last name of the students – for example '**Exercise4_Kirk_Spock.zip**'.
2. As with previous assignments, the submitted zip file should contain**only files needed to build the project and run it.** In other words, the project files, the sources and headers and the resource files (textures, models…) - no intermediate files or directories (such as sdf, obj and others) should be included – review the zip files I created for the book projects as an example.
3. A working release version of your application (.exe file) should be included in the zip and should not rely on any file on my computer to be able to run.
4. **The zipped root directory should contain all files and directories and match the exercise name as per point 1!**

Submission Date:

Sunday, March 29th 2015 at midnight

Overview

This exercise is based on the previous project and will further enhance the existing material and shaders. As before, reference for how to connect the shader, pass parameters and texture can be found in the course text book. I advise you to go briefly over it during the implementation.

Like in the previous project you need to use the following DirectX 9c functions:

- D3DXCreateBox
- D3DXCreateSphere
- D3DXCreateCylinder
- D3DXCreateTeaPot
- D3DXCreateTorus
- Cone (how do you create a cone using the above functions?)

Goals of the exercise:

- Enhance your knowledge of shaders rendering
- Get familiar with more advanced technique (reflection and normal map)
- Get familiar with D3DXMESH class data structure.
- Support Texture and the advance shading of models.

Implementation Specifications

1. Enhance the **BaseMaterial** class you implemented in your previous exercise:
    1. Your previous class should be the parent class of the new material (based on the BaseMaterial class provided in the past.
    2. The class should hold a pointer to the effect file and connect to the effect parameters
    3. During render the class will pass all required data to the shader.
    4. The class pointer should be held by the **BaseObject3D** class so that each object can have its own material with possible different parameters.
    5. During its render, the base model will call the material with all needed data (such as matrices, viewer location, light, colors, textures…) so that the material will set it to the shader prior the render of the model data.
    6. Add the following attributes to each Vertex (you need to enhance the vertex declaration accordingly – review the book for examples):
        1. Texture coordinates
        2. Normal
        3. Tangent
        4. [Bi-normal] – optional

1. Write a short one page documentation (Word or PDF) that describes your implementation
2. Create the effect file that will implement the shading model as explained in class and in the book – **the shader(s) must contains comments explaining each component!**

UI Controls

1. Camera control – we will be now be moving to 'examine mode' camera:
    1. Model should be placed at the (0,0,0) of the world
    2. Mouse movements should control rotation of camera around the model
    3. Mouse wheel should control radius / distance from the model
2. Lighting equation should work perfectly with and without texture, normal mapping and environment reflection.
3. Reflection component should be added as a blend in the specular component
4. Normal mapping should affect both diffuse and specular lighting components.
5. Normal mapping component should have coefficient to how strong it is blended in – think how to do this!
6. The coefficients for blending Ambient, Diffuse and Specular should be 0.2, 0.65 and 0.15 respectfully.
7. UI choice/help should be written at the top left side of the screen (Key, operation, value)

UI keys:

- 'W' - Switch between Solid render and Wireframe
- 'T' - Switch texture on/off
- 'O' - Switch between Objects in cyclic order
- 'R' - Switch environment reflection on/off
- 'N' - Switch normal mapping on/off
- '-' / '+' - Blend between the reflection and specular [0..1] with 0.1 steps
- 'A' / 'S' - Controls the strength of the normal [0..1] with 0.1 steps
- '1'..'7' - Controls the specular coefficient [2, 4, 8, 16, 32, 64, 128]

Grading

- **[15%] Exe code runs properly**
- [5%] Clean code standards
- [10%] Camera / model mouse control
- [20%] All UI modes and model fully functioning
- [20%] Normal mapping working and blended correctly
- [20%] Environment reflections working and blended correctly
- [10%] Make me wow extra cool feature(s)!

Good Luck and May the Force be With You