ME/HCI/CS/CprE 557 - Computer Graphics
Assignment 3
Texture Mapping

Rafael Radkowski

Fall 2017

The goal of this assignment is to work with texture maps. You should understand how textures need to be prepared, how the 3D model needs to be prepared, and what graphics card capabilities you have to use to render a texture on the surface of an object.

Problem 1

Create a single plane and combine three images as textures on this plane. Select three images on your own (I do not provide the images). Each images needs to belong to one of the following categories.

1. color gradient: the image should show a color gradient, at least, between two colors.
2. landscape: use a daylight or nightlight landscape pictures.
3. animal / person: take a picture of an animal or a person, face or the entire animal / person.

Figure 1: Select three images. Each image must cover one of the categories.

Implement a multi-texture OpenGL application and combine / blend / merge/ subtract the images from each other.

Work with different blending equations to blend the color components of the three images. Explore **two** different blending modes at least and describe the differences.

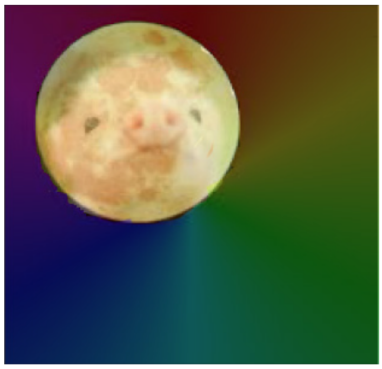

Figure 2: The resulting image: a pig in the moon.

Note:

- Use a scene with complete light, etc. Select the position and orientation as well as the type of all light sources on your own.
- You can use the 3d plane as provided, but feel free to create a new 3D model. Make sure that the light source parameters and the material parameters align with each other.

Problem 2

The objective is to use noise maps to generate a visual effect. So far, images were used as a information source for textures; the color information is used to render the fragments of a 3D object. Instead, using a noise map, use the stored information to displace the (u,v) texture coordinates (see example below).

Implement a multi-texture OpenGL application with two textures (Figure 3): an image of you choice and a normal map image (Ask me if you cannot create one on your own).

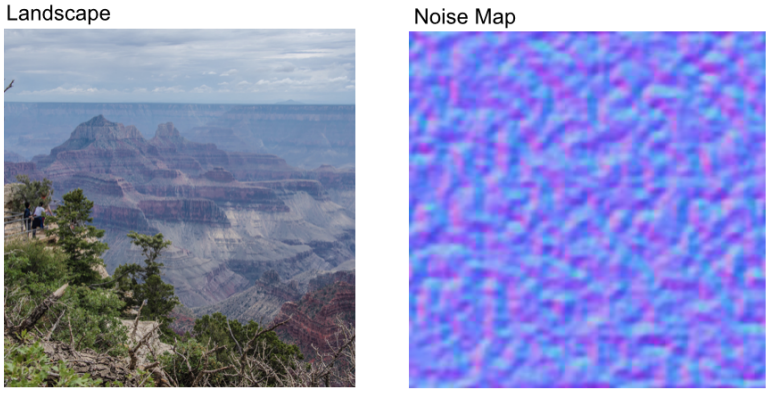

Figure 3: Left: a landscape image, right: a normal map.

Use the information from the noise map (normal map), interpret them as displacement vectors, and change the texture location from which you select the color information for your fragment.

You can start with the following fragment shader code example, which shows such a displacement operation.

```
uniform sampler2D colorMap;
uniform sampler2D noiseMap;

in vec2 texCoord;

void main (void)
{
        vec2 noiseVec;
        noiseVec = normalize(texture(noiseMap, texCoord));
        noiseVec = (noiseVec * 2.0 - 1.0) * 0.035;
        gl_FragColor = texture(colorMap, texCoord + noiseVec);
}
```

Deliverable

The following deliverables must be submitted via git:

- Upload your solution (source code) into via Blackboard.
- Upload screenshots showing your solution.

Due data: Friday, Nov. 3rd, 2017, 8:00 pm

Grading

- P1,2: Correct configuration of your image data (width, height, color model, etc.), 1pt
- P1: Correct implementation of texture coordinates, 1pt
- P1: Correct implementation of texture objects code for three textures, 1pt
- P1: Correct implementation of a glsl shader program that combines three textures to one, 1pt
- P1: Rendering all images using different blend modes, 2pt
- P2: Correct implementation of a noise map texture object, 1pt
- P2: Correct implementation of the glsl shader code for applying a noise map, 1pt
- P3: Rendering of texture on a plane with a visual effect created by a noise map, 2pt