Modeling in Computer Graphics

Exercise Course

20 Juni 2018

Till Niese – till.niese@uni.kn

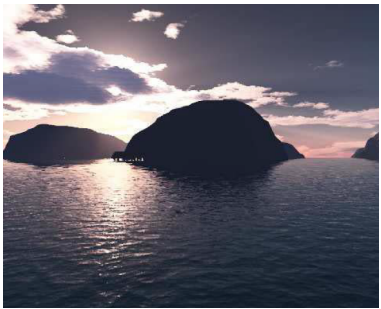
Universität Konstanz

graphics.uni.kn

Exercise 4

Water-Animation and Reflection

Create an animated water surface with reflection and refraction.



Due date **11.07.2018**.

Exercise 4

Task: Water-Animation and Reflection

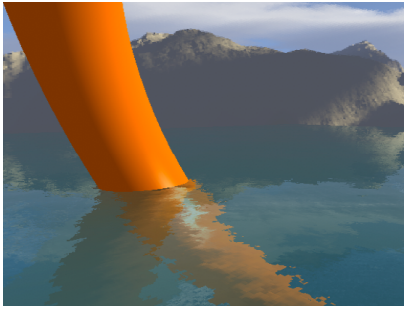
- Create a large grid for the water surface and place it inside of the SkyBox
- Refraction and reflection with fresnel equations
- Perturbation with bump map.

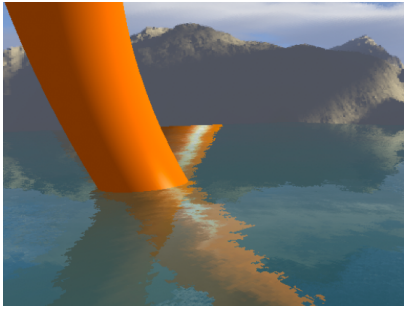
Exercise 4

Reflection

Render the scene from the current camera position (without water)

- ”Mirror” the scene: (1, -1, 1)
- Use framebuffer object to render to a texture: http://www.opengl.org/wiki/Framebuffer_Object
- Use this texture in water shader for reflection color



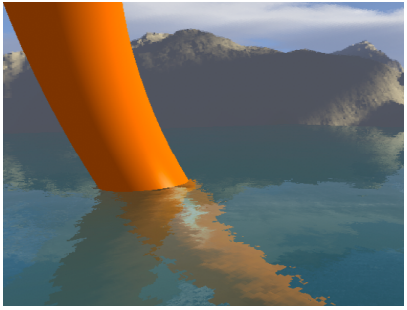


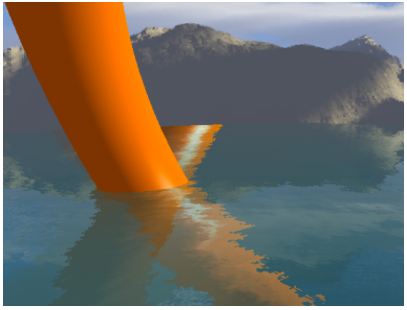
Exercise 4

Refraction

Render the scene from the current camera position (without water)

- Displace the scene to mimic air/water refraction: (1, 1/1.33, 1)
- Use framebuffer object to render to a texture: http://www.opengl.org/wiki/Framebuffer_Object
- Use this texture in water shader for refraction color





Exercise 4

Depth map for simulated water depth

- Render again without water to framebuffer object
- Use depth attachment to compute ground-water distance in your water shader

Exercise 4

Perturbation

”Displacement” map

- Generate noise texture (Perlin) or generate via Terragen
- Displace this texture via timer function for simple sea animation
- Use this texture in your water shader

Exercise 4

FBO Usage

```
m_fbo = std::make_shared<cgfw::gl::FrameBuffer>();

m_colorAttachment = std::make_shared<cgfw::gl::Texture>(GL_TEXTURE_2D);
m_colorAttachment->setTexureWrapS(GL_CLAMP_TO_BORDER);
m_colorAttachment->setTexureWrapT(GL_CLAMP_TO_BORDER);

m_depthAttachment = std::make_shared<cgfw::gl::Texture>(GL_TEXTURE_2D);
m_depthAttachment->setTexureWrapS(GL_CLAMP_TO_BORDER);
m_depthAttachment->setTexureWrapT(GL_CLAMP_TO_BORDER);
```

Exercise 4

FBO Usage

```
m_colorAttachment->image2D(0, GL_RGB, width, height, GL_RGB);
m_depthAttachment->image2D(0, GL_DEPTH_COMPONENT32, width, height,
    GL_DEPTH_COMPONENT);

m_fbo->colorAttachment(0, m_colorAttachment);
m_fbo->depthAttachment(m_depthAttachment);

m_fbo->bind();
// render the scene here
m_fbo->unbind();

m_colorAttachment->generateMipmap();
m_depthAttachment->generateMipmap();
```