Racz Adrian-Catalin
30431

UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA

Project Documentation

1. Scenario

a. Scene description

The scene represents a playground/sandbox for testing OpenGL features and algorithms. It is composed out of a few objects: the ground, a nanosuit, a teapot and a skybox; all of them are textured, while the nanosuit and teapot are also interactive (they can play their animations or be rotated and moved around). A source of light is also present, lighting the objects and creating shadows. The light source can be rotated either around the scene to cast shadows from different angles, or bellow the ground and back above to simulate a day and night cycle, making the scene progressively darker as the sun sets and brighter as the sun rises. There is also an optional fog filter, triggered from a button. The camera is free floating, able to move and rotate anywhere to get better points of view. The animations include the teapot rocking from side to side, the nanosuit walking back and forth between 2 positions, including turning on direction change as to always walk forward, and the light is simulating the sun by going from east to west then underground and back up again.

b. Functionalities

The user is able to control the camera and move it wherever he likes and rotate it for any desired angle (3 dynamic axys movement, the front is always where the camera is pointed at and the sideways axys is always perpendicular on front and up vectors plane, 2 axys rotation). The animations can be started and stopped by two different buttons and the scene or camera can be reset to the initial, idle, state. The 2 main objects (teapot and nanosuit) can be moved and rotated at will, however the selected object must be chosen first as there weren't enough buttons. The fog can be triggered on or off and the light can be rotated either around the X axys or the Z axys (this also reuses the object selection buttons). Also, the window is fully resizable, adapting the projection matrix to the new height and width.

2. Implementation

The renderScene() function is the main function that deals with drawing the objects. First it checks if the animations are enabled and progresses them if they are. Then it prepares the shadow map created from the light's viewpoint. Our light is directional, therefore an orthographic light projection will suffice for the light space matrix. After sending the matrix and binding the shadow map (depth map), the shadows are done and it moves to the renderObjects() function, which draws each individual object, applying their models.

The shaders used are the skyboxShader, shadowShader and the basicShader (all of them with frag and vert shaders). The shadow shader is used for drawing the shadows, but it is simplistic as it only calculates the position from the light's viewpoint in the vertex shader and does nothing in the fragment shader. The skyboxShader is not too complicated, it only deals with applying the skybox texture to the cube that encompasses the entire scene. Additionally, the fragment shader also changes the color of the texture to grey if the mist is enabled, or darker and darker as the sun sets.

The basic shader is the one used for all objects. It deals with texture mapping, the light, shadowing and fog effects. The light is implemented as in the laboratories, by computing the specular, ambient and diffuse lights and applying them to the color. The shadows are also implemented as in the laboratories; it compares the shadow map and if the position is in the shadow, it darkens the fragment. The fog is implemented by computing the distance from the camera in the vertex shader, and, using an exponential formula, a density and gradient values, it calculates the visibility. Based on the visibility factor, in the fragment shader a grey color is mixed with the normal color, less intense the higher the visibility is.

The camera movement has been fully implemented in the Camera.cpp. The movement uses the front direction, right direction and up direction of the camera and, also, applies the translation to the camera target, not only on its position so that the camera always remains facing the same way.

The skybox is just a cube with a texture on all inside faces, with the center in the camera, that encompasses the entire scene.

There is an extra addition tied to the rotation of the light. When it is the highest, the colors of the objects and the skybox are the brightest, but when it starts to set, by using a light brigthness coefficient, we turn the colors darker and darker, and when it starts to rise again, we turn them back brighter and brighter.

The resizeable window is implemented in the windowResizeCallback() function, that updates the window size of our window object and then recomputes the projection matrix.

3. User controls

- WASD and RF: for camera movement, maintains the same view direction by also moving the camera target (forward, left strafe, backward, right strafe, up, down)
- The arrow keys: for camera rotation
- YHUJNM: object movement on the 3 axis (Z axis, Y axis, X axis)
- QEIK: object rotation around 2 axis
- ZX: reset objects, reset camera
- VB: start animations, stop animations
- 12: select active object as either the Teapot or the Nanosuit
- P: rotate light (around which axis is chosen from the object selection)
- C: fog enable/disable

4. Bibliography

The laboratories

https://learnopengl.com/Advanced-Lighting/Shadows/Shadow-Mapping

https://www.mbsoftworks.sk/tutorials/opengl4/020-fog/

https://www.youtube.com/watch?v=qslBNLeSPUc