02561 COMPUTER GRAPHICS DTU COMPUTE

Worksheet 5: Rendering a triangle mesh

Reading	RTR: Chapter 16 WebGL Programming Guide: “Load and Display 3D Models”
Purpose	The purpose of this set of exercises is to load a triangle mesh that models something more interesting than a box. Once the model is loaded, we will apply lighting and shading using the techniques from Worksheet 4. In the following, we will ask you to put your solutions on your student homepage. This works for all browsers and it is useful for you to know how to put your WebGPU programs online. However, if you would like to test code locally, some options are listed here: https://courses.compute.dtu.dk/02561/test-locally.html
Part 1	We have now reached a point where we would like to load external content into our WebGPU applications. This is troublesome (due to browser security settings) if our application is not running as an actual webpage on a webserver. You should therefore now place your application on a “secret” webpage. A webpage is secret if there are no public links to the page, as it is then not searchable. Placing your WebGPU application on your DTU Student Homepage: • Get started on your student homepage using the gbar tutorial: https://www.gbar.dtu.dk/index.php/faq/50-homepage If the shell script that should create a link to your public_html folder is out of order, use the command *pwd* to get the path of your home directory. Then use this path to find the path to your homepage folder. As can be seen in the tutorial, the home page path should be of the form */www/xx/x/public_html*. Once you have located your homepage folder, you can create a link to it in your standard home directory using the command *ln -s /www/xx/x/public_html public_html* • Get access to the files of your student homepage using SCP or likewise: https://www.gbar.dtu.dk/index.php/faq/78-home-directory https://www.gbar.dtu.dk/index.php/faq/25-winscp • Put a file called index.html in your public_html folder. This ensures that the files in this folder are not browsable (files in subfolders will conveniently still be browsable). • Put the JavaScript library files in a subfolder on the server and place a previous exercise solution on the server in another subfolder. If you make no public links to this subfolder, you can use it as your secret folder. • You have completed this part when one of your previous exercise solutions can be loaded as a webpage in a browser.

Part 2	Create a nice 3D object using a modeling tool such as Blender or Maya and export it as a triangle mesh in Wavefront OBJ format. The modelled object must be more interesting than a box. The Blender monkey called Suzanne is an option. If you are absolutely at a loss with respect to modelling a 3D object and exporting it as an OBJ file, use the teapot uploaded to DTU Learn. [This option is a quick way to move on, but not a full solution for this part.] Upload the OBJ file and the associated MTL file (if used) to the server, so that your WebGPU application is able to load it.
Part 3	The next step is to load the OBJ file. A method for loading and displaying such files is given in the section "Load and Display 3D Models" from the WebGL Programming Guide (available on DTU Learn). The part of the code that parses the OBJ file is in the file OBJParser.js, which we have uploaded to DTU Learn. Place OBJParser.js on the server together with your other JavaScript files and use the `async` and `await` keywords together with the function `readOBJFile` to load the triangle mesh from the OBJ file. Once data is available, pass it to WebGPU buffers, set up the camera, and draw your 3D object as an indexed face set using a simple set of shaders.
Part 4	Set up a light source and use your shaders from Part 5 of Worksheet 4 to shade the object using Phong shading and the Phong illumination model. Explain how you obtain and use surface normals and explain how this relates to the surface smoothness when you are rendering a triangle mesh.