02561 COMPUTER GRAPHICS DTU COMPUTE

Worksheet 5: Rendering a triangle mesh

Reading	Angel: Section 4.6. 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 WebGL programs online. However, if you would like to test code locally, it is possible to run an insecure version of some browsers. You can for example test programs that load content from external files in Chrome by closing all running instances of Chrome and running it as follows: `chrome.exe --allow-file-access-from-files` If you do this, it is important that you close Chrome and reopen it without this insecure setting before using it for regular internet browsing.
Part 1	We have now reached a point where we would like to load external content into our WebGL 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 WebGL application on your DTU Student Homepage: • Get started on your student homepage using the gbar tutorial: http://gbar.dtu.dk/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: http://gbar.dtu.dk/faq/78-home-directory http://gbar.dtu.dk/faq/25-winscp • Put the JavaScript library files in a subfolder on the server and place a previous exercise solution on the server. Preferably also in a subfolder. [Please remember to keep your solutions secret.] • You have completed this part when one of your previous exercise solutions can be loaded as a webpage in a browser.

Worksheet 5: Rendering a triangle mesh

Part 2	Create a nice 3D object using a modeling tool such as Blender or Maya or Google SketchUp 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 a nice 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 CampusNet. Upload the OBJ file and the associated MTL file (if used) to the server, so that your WebGL 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 CampusNet). The part of the code that parses the OBJ file is in the file OBJParser.js, which we have uploaded to CampusNet.
	
	Place OBJParser.js on the server together with the other library files and use Listing 10.18 from the WebGL Programming Guide to load the triangle mesh from the OBJ file.
	
	Since browser programs (like the one you are developing here) do not wait for data to load, you need to wait for the data before using it in the render function that you are calling for every frame. Use the following lines of code to wait:
	
	`if (!g_drawingInfo && g_objDoc && g_objDoc.isMTLComplete()) {`
	`  // OBJ and all MTLs are available`
	`  g_drawingInfo = onReadComplete(gl, model, g_objDoc);`
	`}`
	`if (!g_drawingInfo) return;`
	
	Once data is available, the `onReadComplete` function passes it to WebGL buffers. Get this function from Listing 10.21 of the WebGL Programming Guide.
	
	Set up the camera and draw your 3D object as an indexed face set using a simple set of shaders. The draw call for the loaded OBJ file is
	`gl.drawElements(gl.TRIANGLES, g_drawingInfo.indices.length,`
	`  gl.UNSIGNED_SHORT, 0);`

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.