INTERACTIVE 3D GRAPHICS – PROJECT 2

STUDENTS

Soprano Michael – 112151 – soprano.michael@spes.uniud.it

Perazza Giuliano – 112461 – perazza.giuliano@spes.uniud.it

GOALS

1. Use lights and shaders to present the graphs in a more appealing way: **SATISFIED**
2. Use animations to "grow" the graph, i.e. start with an empty graph and "grow" elements as the graph is created (the animation should last just a few seconds): **SATISFIED**
3. Visualise labels (of rows and/or columns, and numeric values on the axes, where applicable): **SATISFIED**

REQUIREMENTS

1. Choose a suitable number and type of lights so that the graph is properly illuminated (minimum 3 lights): **SATISFIED**

We used three spotlights to light the charts, and an additional ambient light

2. Try to use materials in a consistent way; for example, if you choose metal-like materials, use different kind of metals for the different elements in the graph: **SATISFIED**
3. You need to provide two choices of materials for each graph in the menu, where one does not uses three.js materials (except the ShaderMaterial) and works with shaders written by you (your shader should take into account specular and Fresnel effects); the other one uses three.js materials with shadow mapping: **SATISFIED**

SUGGESTIONS

1. For drawing labels, you can use various methods, from 3D text (TextGeometry object in three.js) to HTML elements drawn on top, or dynamically constructing textures from HTML canvas elements (see e.g. http://stemkoski.github.io/Three.js/Texture-From-Canvas.html). Choose the option that will give you the best results.

To implement the labels, we decided to use the TextGeometry object, as we did for the instruction in the main page of the application.

2. You can re-implement highlighting by using a different shader, or by adding a light.

We used another shader to reimplement the highlight, instead of adding more lights.

EXTRA CAPABILITIES

1. There are a diffuse and a specular texture to create three metal-like planes for the bar and area chart, and one plane for the pie chart.

SHADING EQUATION

Our custom BRDF used in the shading equation is the Blinn-Phong one, which models the specular reflection of glossy materials using the idea of microfacets. The equation is evaluated per vertex, inside the vertex shader; we take account of the contribute of every light, by applying the Blinn-Phong's formula, and also considering an ambiental term, which is a coefficient that multiplies the diffuse color vector of the surface, and the Fresnel effect, which models the amount of light reflected considering the interaction between two substances. The output of the vertex shader is the value of the out radiance. In the fragment shader, the out radiance is simply set as the color of the fragment itself. The other two parameters used in the equation are the biradiance β_i, computed for every light, which is the power entering in the surface with direction l_i , and the dot product between l_i direction and v direction, which is the exit direction of the radiance; this product models the observed area of the surface.

In the following, a brief high-level description of the formula of the equation:

$$OUTRADIANCE = A + \left((\beta 1 * |l1 \cdot v| * BRDF(l1, v)) + (\beta 2 * |l2 \cdot v| * BRDF(l2, v) + (\beta 3 * |l3 \cdot v| * BRDF(l3, v))\right)$$