Lab 2

TDT4195: Grunnleggende Visuell Databehandling

Jonas Myrlund

September 16, 2013

I Change RenderScreen5

To get the top left corner involved, I changed the vertex matrix to the following (note the awesome new general shape):

```
static const GLfloat g_vertex_buffer_data[] = {
    -1.0f,   1.0f, 0.0f,
    1.0f,   -1.0f, 0.0f,
    -0.75f,  1.0f, 0.0f,
};
```

In order for it to turn blue, I changed the SimpleFragmentShader to the following:

```
#version 330 core

// Ouput data
out vec3 color;

void main()
{

  // Output color = blue
  color = vec3(0,0,1);

}
```

II Create RenderScreen6

We'll reuse the shader from RenderScreen5, and simply extend the `g_vertex_buffer_data` array with the required vertices.

To make the scene load, I also had to perform some straight-forward changes around the project, but I won't detail them here.

The matrix now looks like this:

```
static const GLfloat g_vertex_buffer_data[] = {

    // Scene 5
    -1.0f, 1.0f, 0.0f,
    1.0f, -1.0f, 0.0f,
    -0.75f,  1.0f, 0.0f,

    // Scene 6
    -1.0f, -1.0f, 0.0f,
    0.0f,   0.0f, 0.0f,
    0.0f,  -1.0f, 0.0f,

    -1.0f, 1.0f, 0.0f,
    0.0f,  0.0f, 0.0f,
    0.0f,  1.0f, 0.0f,

    1.0f, 1.0f, 0.0f,
    1.0f, -1.0f, 0.0f,
    0.0f, 0.0f, 0.0f,

};
```

And the relevant code in RenderScreen6 looks like this, working from an offset of 3 vertices:

```
glDrawArrays(GL_TRIANGLES, 3, 9);
```

Otherwise, RenderScreen6 looks just like RenderScreen5.

III Minor code tasks

1 Center window

The x position is half the window's width to the left of the middle of the screen. Likewise for y.

```
int iWindowWidth = 512;
int iWindowHeight = 512;

glutInitWindowPosition( iScreenWidth / 2 - iWindowWidth / 2,
                        iScreenHeight / 2 - iWindowHeight / 2 );
glutInitWindowSize( iWindowWidth, iWindowHeight );
```

2 Render a torus

Replacing the rendering call in RenderScreen1 with:

```
glutSolidTorus(10.0, 20.0, 100, 200);
```

3 Explain why the Scene 1,2,3 and 4 are displayed wrong after the user enable scene 5

The `glUseProgram(shaderProgramId)` call says that we want to use the given shader, as we do in scene 5. However, the reset call, `glUseProgram(0)` is never called, so the shader from 5 still applies.

Calling `glUseProgram(0)` whenever switching between scenes solves this problem.