Lab 2: 3D Modelling

3D Computer Graphics

Introduction

Import the archive file `3DCG_Lab2.jar` into Eclipse by selecting

```
File > Import... > General > Existing Projects into Workspace
            > Next > Select archive file > Finish
```

If you have set up JOGL correctly in the previous lab, you can simply add the user library jogl-2.0 to this project as follows: right-mouse click on the project's name in the Package Explorer window and select

```
Build Path > Add Libraries ... > User Library > jogl-2.0.
```

Exercise 1

The `Point` and `Vector` class in the `util` package represent a point and vector, respectively, in a 3D space.

a) Have a look at the methods of these two classes so that you will remember them later on when you need them.

b) Study the implementation of these methods in detail and find the three bugs in the code.

Exercise 2

Have a look at the file `wineglass.txt` in the `resource` folder. It has the same file format as discussed in the slides of this lab.

a) How many faces does this 3D model have? And how many normal vectors?

The aim of this exercise is to render (= visualize) this 3D model.

b) Study the implementation of the `Face` class in the `mesh` package. Make sure that you understand the aim of each instance variable and method of this class.

c) Make sure that you understand how the `Mesh` and `Face` class work together to represent a 3D object.

d) Implement the `readFile` method of the `Mesh` class which reads the 3D model data from the given file. One may assume that this file has the format as explained in the slides of this lab. This method should store the data in the appropriate data structures. Make use of the `java.util.Scanner` class to help you reading in the file data.

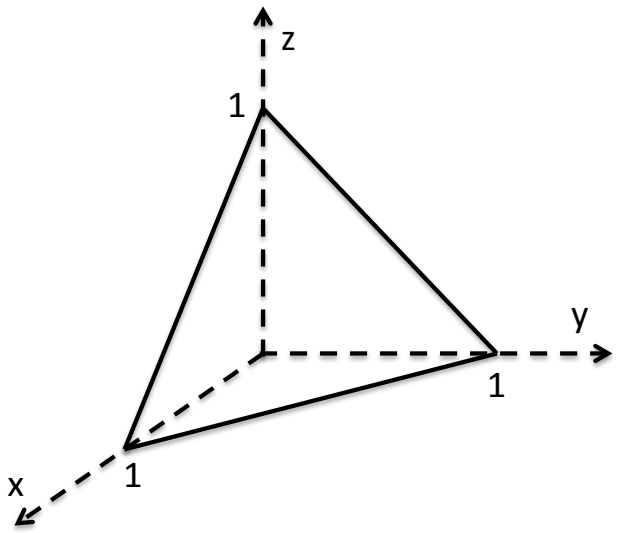
e) Have a look at the `draw` method of the `Mesh` class. It has a loop over all faces. For each face it tells OpenGL about the coordinates of the normal vectors and points corresponding to its vertices (with the `glNormal3f` and `glVertex3f` OpenGL method calls). Note however that the coordinates of the normal vectors and vertices are not properly initialized. Can you fix this?

f) Look at the `display` method of the `App1` class in the `app1` package. This method is responsible for drawing the virtual scene. You do not have to understand the whole implementation. Just note that the `draw` method of a `Mesh` object is called. This `Mesh` object, however, is not properly initialized. Can you fix this so that this application will render the wineglass?

g) Run `App1` to check your solutions to the previous questions.

Exercise 3

The figure below shows a tetrahedron.



a) Create a new file `tetrahedron.txt` in the `resources` folder with the correct data for this tetrahedron according to the file format discussed in the slides of this lab.

b) Change one line of code in the `App1` class to render your tetrahedron.

c) Run `App1` again. Do you get the expected result?

d) The tetrahedron is not centered on the screen. This is due to the camera which is not properly aimed at the tetrahedron. Look at the `setCamera` method of the `Renderer` class in the `util` package. By commenting out the last line of this method implementation and using the last but one line instead, the camera position is changed in a way which should put the tetrahedron more centered on the screen. (We will discuss the camera position and orientation in detail in the next lab.)

Exercise 4

The data below are the polygonal mesh representation of a pyramid. The data format corresponds to the file format we used to store a polygonal mesh of a 3D object. However, the data are not complete. At the end one or more lines are missing. Can you reproduce this missing data?

```
5 5 5
-5 -5 0 5 -5 0 5 5 0 -5 5 0 0 0 5
0 0 -1 0 -0.707 0.707 0.707 0 0.707
0 0.707 0.707 -0.707 0 0.707
3 3 0 4 4 4 4
3 0 1 4 1 1 1
```