Computer Graphics - 2016/2017

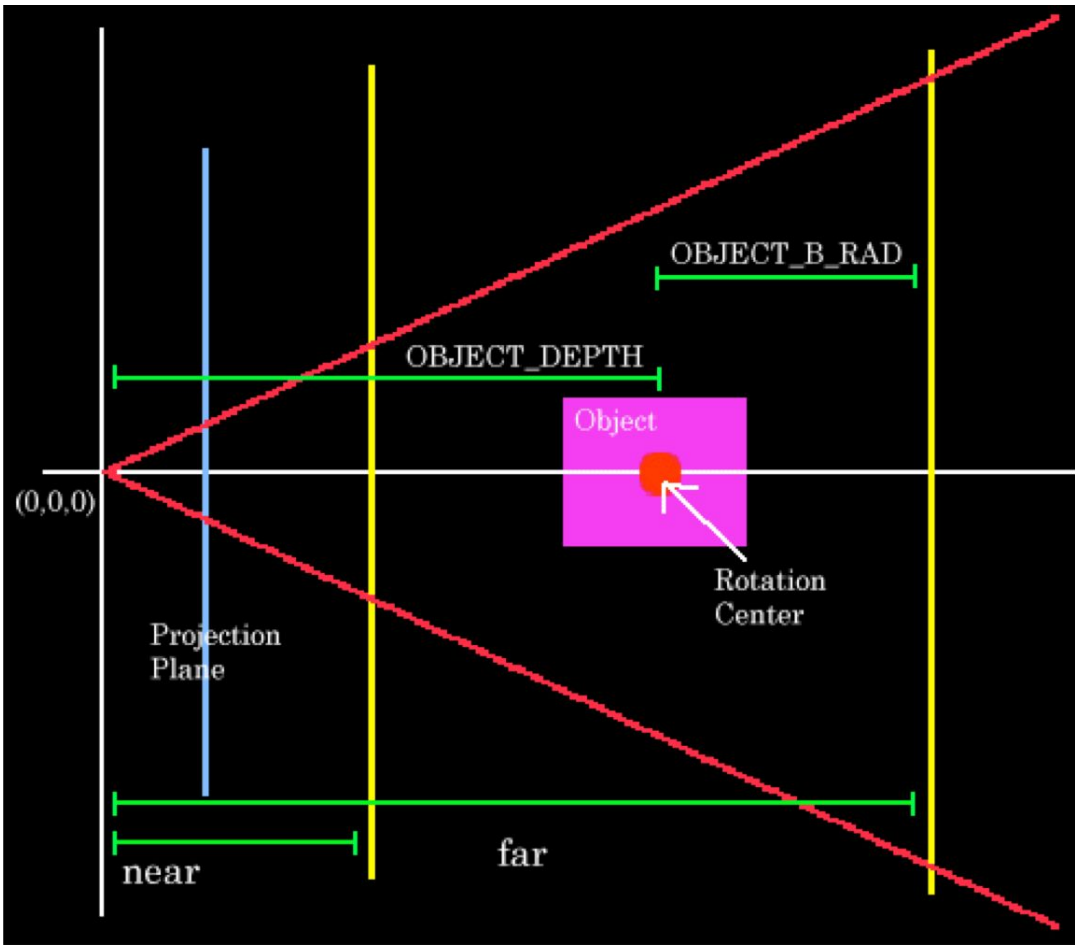
Exercise 2

Last update: 2016 Dec 06, 21:55

In this exercise you will create an application for interactively viewing 3D objects. The program supports rotating an object, changing its position and zooming in and out. You will use OpenMesh to represent the 3D objects and to read them from disk (no parser to write in this exercise).

In particular, you will read a file describing a 3D object (using the OpenMesh library) and view the object defined in the file using an arcball interface, as explained in class. Make sure to go over the tirgul slides and understand how this interface works, before you start working.

After reading the object description, you will display a perspective view of the object, arranged as shown in the following diagram:



1. Choose some value for the OBJECT_DEPTH constant. Also choose OBJECT_B_RAD such that OBJECT_DEPTH - OBJECT_B_RAD > 0. Set near to be OBJECT_DEPTH - OBJECT_B_RAD and far to be OBJECT_DEPTH+OBJECT_B_RAD. When loading an object from a file make sure that you estimate the scaling factor so the object will fit the defined frustum. Start with the field of view of 30 degrees in the perspective projection. Make sure your frustum's aspect ratio is equal to that of the display window.

2. **Using OpenMesh**: In the slides from the class you can find example code that uses OpenMesh to accomplish several simple tasks.
Link to OpenMesh documentation: here.

2.1. **Makefile:** You should add the following lines to your makefile:

```
#OpenMesh definitions
CG_HOME = /cs/course/2013/cg
OM_DIR = $(CG_HOME)/OpenMesh
OM_LIBS = -L$(OM_DIR)/lib -lOpenMeshCore
OM_INCLUDE=  -I$(OM_DIR)/include
```

Then, add `$(OM_INCLUDE)` to the compiler flags and `$(OM_LIBS)` to the linker flags.

Update Dec 6 2016: You should also add this `-D_GLIBCXX_USE_CXX11_ABI=0` to your compiler flags, and don't use C++11 features.

2.2. **Environment configuration**: OpenMesh is compiled as a dynamic library. In order to run it you will have to set the environment variable LD_LIBRARY_PATH to include the location of the libraries. Run this (one line) in order to set this variable correctly:

```
setenv LD_LIBRARY_PATH
/cs/course/2013/cg/OpenMesh/lib/:/cs/course/2013/cg/bimage/:/cs/cours
e/2013/cg/linear/
```

You also need to run this script before running the school solution.

3. **Displaying the mesh:** The mesh may be drawn in one of two modes (switch between them with ‘W’ or ‘w’):
3.1 Wire-frame mode: Only the edges of the mesh are drawn.
3.2 Full mode: Draw filled triangle faces.
Each mesh vertex should be assigned a color that visualizes its position in 3D (x in red, y in green, and z in blue). Make sure that the mapping from (x,y,z) to (r,g,b) is such that the color channel values are in the range [0,1]. Use the object bounding box for that purpose. The fragment shader should then assign each fragment the color interpolated from the vertices.

4. Keyboard Operations:
R / r : Reset to the initial view of the model: rotation, translation and zoom
Q / q : Quit the program. Simply call exit().
W / w : Toggle between wireframe and full mode.
P / p: Toggle between perspective projection (default) and orthographic projection. You should make sure that the orthographic projection is “compatible” with the current perspective view, i.e. it displays the object at roughly the same size, and responds similarly to zooming in and out.

5. Mouse operations:
Mouse events should be interpreted as following:
Left button: this button will be responsible for rotating the object as described in the arcball notes.
Right button: will move the object on the plane parallel to the viewing plane. As the mouse moves on a plan (the window) - i.e. defines a 3D motion, we’ll want the object to do the same motion on a plane parallel to the viewing plane.

Middle button: will zoom in and out. This will be done by opening and closing the angle of the field of view.

Important Note: The total transformation of the object by the Arcball should be the accumulation of completed transformations done by the user so far + the current transformation (assuming the the user is clicking the mouse button). Completed Transformation is defined as the final transformation between the mouse down event and the mouse up event. The current transformation is the transformation performed by the user while dragging the mouse (motion event). Do not accumulate transformations done during motion event, but display them. Accumulate only when the mouse button is released.

General:

- Use double buffering, `RGBA` and `DEPTH` modes for initializing the GLUT display mode (`GLUT_DOUBLE | GLUT_RGBA | GLUT_DEPTH`).
- You should also enable hidden surface removal by calling `glEnable(GL_DEPTH_TEST)` somewhere in your OpenGL initialization code.
- At the beginning of your display function clear the depth buffer bit in addition to the color buffer bit as follows: `glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);`
- Usage `ex2 <input-file>`
  Since OpenMesh reads the mesh for us, we support all the formats that OpenMesh supports. You can use the sample models from here.

Tips:

- Add the program's feature one by one:
  Start by reading and displaying a static OpenMesh object. Then implement the Arcball interface.
- Think carefully about the order of transformations: when adding the current transformation to the accumulated transformation - in which order should it be applied?

Submission

Include the following in your submission

1. A Readme.txt file, that includes:
   - Your id and login
   - Your partner's id and login
   - A brief description of your implementation, which piece(s) of code you started from and what changes you made to it (/them).
   - All files that are required for compilation of your solution with a single 'make' command, and the shaders necessary to run it.

Pack all files as a single zip file named by the following pattern: ex1_<your 9 digits id>_<your_username>_<your partner's 9 digits id>_<your partner's 9 digis login>.zip (e.g. '`ex2_123456789_mylogin_987654321_myfriendlogin.zip`').

School Solution:

You can find the school solution here. Before running the school solution, run this (as stated above):

```
setenv LD_LIBRARY_PATH
/cs/course/2013/cg/OpenMesh/lib/:/cs/course/2013/cg/bimage/:/cs/course/2013/cg/linear/
```

Deadline:

You have to submit your solution (via the course's moodle webpage) no later than Thursday 20/12 at 23:45.

Late submission will result in 2^(N+1) points deduction where N is the number of days between the deadline and your submission (rounded up, the minimum grade is 0, friday and saturday are excluded).

Good luck and have fun!