CT5510: Computer Graphics

OpenGL: Setup 3D World

BOCHANG MOON

Prerequisite for 3D World

- Understanding on basic mathematical background, transformations, and spaces
  - Pixels, raster image, …
  - Vector, matrix, …
  - Model, viewing, and projection transformations
  - Object, world, eye, canonical view, and screen space

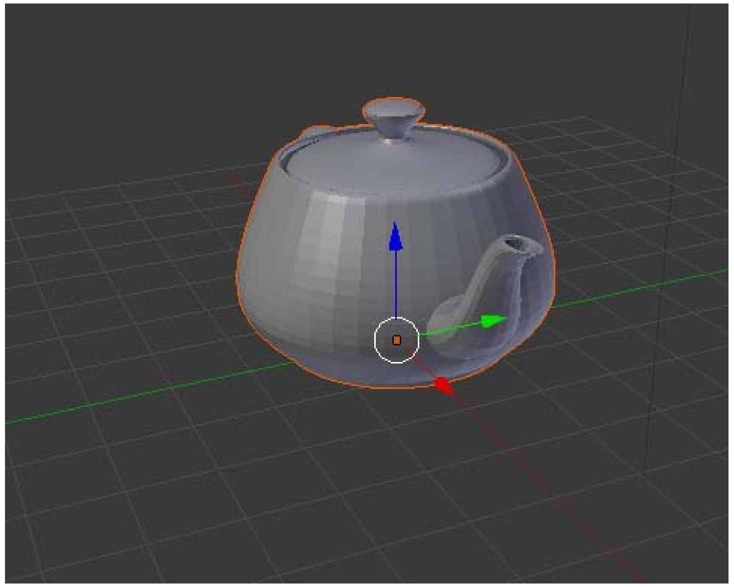
3D Model

- Definition of a model (or object) in 3D
  - Vertex
  - Normal (optional)
    - Q. Why do we need to use the vertex normal?
  - Texture coordinates (optional)
  - Face (usually triangles)
  - etc.

- File format for 3D models
  - You can make your own format only for your program.
  - Common formats
    - 3DS, MAX, ply (Stanford graphics lab), obj (Wavefront), etc.
  - Simple formats
    - ply and obj are quite simple formats
    - In this course, we will use “obj” format as this can be used in most rendering engines.

Example: .obj File Format

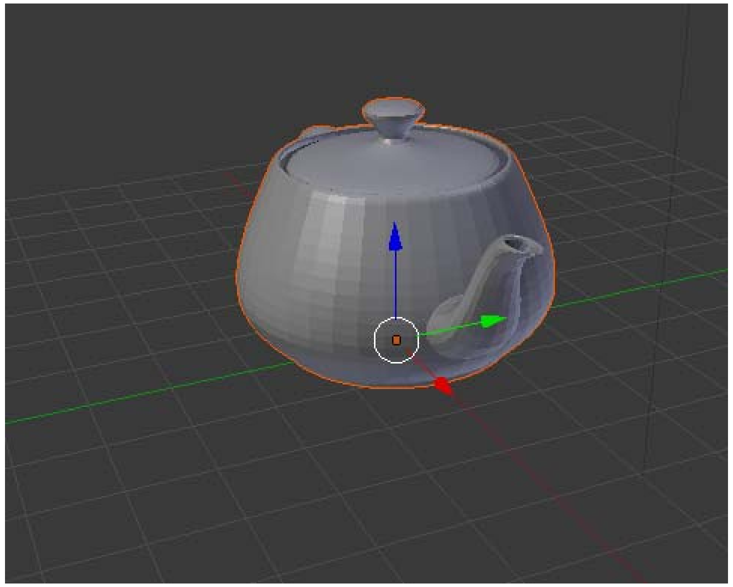
- Artists (or you) can design 3D models in some modeling tools (e.g., blender).
  - Out of scope…
- Most modeling tools allow us to store the models in .obj format.
  - For your homework, .obj files will be given.



Example: .obj File Format

```
v -3.000000 1.800000 0.000000
v -2.991600 1.800000 -0.081000
v -2.991600 1.800000 0.081000
v -2.989450 1.666162 0.000000
v -2.985000 1.921950 0.000000
v -2.985000 1.921950 0.000000
v -2.981175 1.667844 -0.081000
v -2.981175 1.667844 0.081000
v -2.976687 1.920243 -0.081000
v -2.976687 1.920243 0.081000
v -2.968800 1.800000 -0.144000
v -2.968800 1.800000 0.144000
v -2.958713 1.672406 -0.144000
v -2.958713 1.672406 0.144000
v -2.957600 1.534800 0.000000
v -2.957600 1.534800 0.000000
v -2.954122 1.915609 -0.144000
v -2.954122 1.915609 0.144000
v -2.949693 1.537790 -0.081000
v -2.949693 1.537790 0.081000
v -2.940000 2.019600 0.000000
v -2.935200 1.800000 -0.189000
v -2.935200 1.800000 0.189000
v -2.931958 2.016526 0.081000
v -2.931958 2.016526 -0.081000
v -2.928230 1.545907 -0.144000
v -2.928230 1.545907 0.144000
v -2.925611 1.679131 -0.189000
v -2.925611 1.679131 0.189000
v -2.920870 1.908779 -0.189000
v -2.920870 1.908779 0.189000
v -2.910131 2.008181 -0.144000
v -2.910131 2.008181 0.144000
v -2.904150 1.406137 0.000000
v -2.904150 1.406137 0.000000
v -2.896846 1.410135 0.081000
v -2.896846 1.410135 -0.081000
v -2.896602 1.557869 -0.189000
v -2.896602 1.557869 0.189000
v -2.894400 1.800000 -0.216000
v -2.894400 1.800000 0.216000
```

```
f 2909 2921 2939
f 2939 2931 2909
f 2869 2877 2921
f 2921 2909 2869
f 2819 2827 2877
f 2877 2869 2819
f 2737 2747 2827
f 2827 2819 2737
f 2669 2673 2747
f 2747 2737 2669
f 2567 2575 2673
f 2673 2669 2567
f 2476 2480 2575
f 2575 2567 2476
f 2358 2362 2480
f 2480 2476 2358
f 2158 2162 2362
f 2362 2358 2158
f 1715 1812 2162
f 2162 2158 1715
f 2901 2909 2931
f 2931 2917 2901
f 2863 2869 2909
f 2909 2901 2863
f 2813 2819 2869
f 2869 2863 2813
f 2729 2737 2819
f 2819 2813 2729
f 2663 2669 2737
f 2737 2729 2663
f 2561 2567 2669
f 2669 2663 2561
f 2468 2476 2567
f 2567 2561 2468
f 2350 2358 2476
f 2476 2468 2350
f 2152 2158 2358
f 2358 2350 2152
f 1717 1715 2158
f 2158 2152 1717
f 2903 2901 2917
```



teapot.obj (toy example)
- 3644 vertices
- 6320 faces

Example: .obj File Format

- #: comment line
- v x y z w
  - Vertex coordinates in model space
  - w: optional (default = 1)
- vt u v
  - Texture coordinates (0 <= u, v <= 1)
- vn x y z
  - Normal direction
- f v1 v2 v3
  - v1: index in the vertex list (integer)
- Q. why do they use the vertex index instead of coordinates?

Example: .obj File Format

- f v1 v2 v3
  - v1: index in the vertex list (integer)
- f v1/vt1 v2/vt2 v3/vt3
  - vt: texture coordinate (index)
- f v1/vt1/vn1 v2/vt2/vn2 v3/vt3/vn3
  - vn: normal (index)
- f v1//vn1 v2//vn2 v3//vn3
  - Need empty slash to avoid ambiguity

Loading .obj Model

- Read .obj files from your disk
  - Build a vertex list from each model
  - (optional) Create normal and texture coordinate lists
  - Build a face list
    - In our example, we will use triangles.

- A very simple .obj loader will be given for your assignments.
  - ModelLoader.h & ModelLoader.cpp

Draw Triangles

- For each triangle in a model
  - glBegin(GL_TRIANGLES)
    - For each vertex in a triangle
      - glVertex3d(x, y, z)

      - (optionally)
      - glNormal3d(nx, ny, nz)   // related to shading
      - glTexCoord2d(u, v)   // related to texture mapping
  - glEnd()

OpenGL Display List

- Display list: a set of OpenGL commands that have been stored for later execution

- Once the list is compiled (one time), it can be re-used multiple times.
  - Very efficient for static models

Example of OpenGL Display List

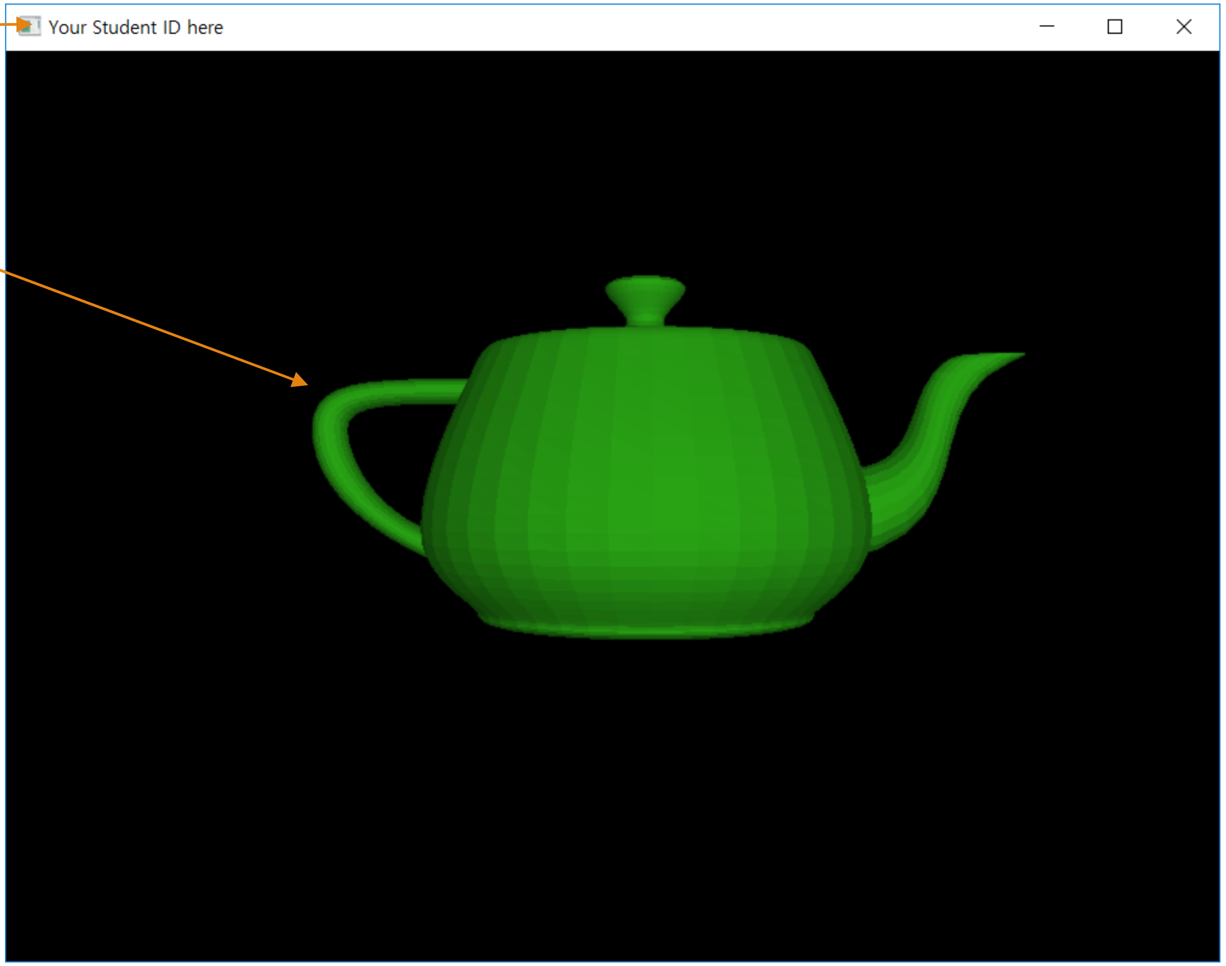
- g_teapotID = glGenLists(1) // create a list and store the ID to the variable
- glNewList(g_teapotID, GL_COMPILE) // Define the set of commands
- glBegin(GL_TRIANGLES)
  - For each vertex in a triangle
    - glVertex3d(x, y, z)

    - (optionally)
    - glNormal3d(nx, ny, nz)   // related to shading
    - glTexCoord2d(u, v)   // related to texture mapping
- glEnd()
- glEndList()

- Draw some models with IDs
  - glCallList(g_teapotID)

Result Image

- Title bar
- Teapot model
  - A set of triangles


Your Student ID here