DigiPen Institute of Technology

CS300

Assignment 1: Loading and displaying an OBJ

Due date:

- Week 4, Friday, 23:59:00
- For late submission, please refer the course outline document.

Description

For your first programming assignment, you are to implement a GUI-driven application with the following requirements/features:

1. Develop an OpenGL + freeGLUT based application that displays an OBJ file. **You may use the framework provided by the TA.**
2. The default display function uses built-in functionality to display a triangle. You must override this functionality to display an *.obj file that will be loaded from the disk. Use the GUI functionality to input the file name.
   - User should select the file from disk using standard file open dialog, or type the file name in an edit box.
   - By default, load the obj file "cube.obj" provided with the framework.
   - Calculate the vertex normals from the face normal in the neighborhood of a vertex. Make sure that the vertex normals account for corner-cases where there may be more than one face in the same plane. Ignore normals that are parallel to any previously accumulated normals in the vertex-normal computation.
   - We will use the provided OBJ files to test your program. Make sure that your OBJ importer is able to read the supplied files.
   - Scale the object so that it completely lies in the range [-0.5, 0.5] in all dimensions, i.e. a cube with unit length sides. Use uniform scaling to achieve this. (What would happen if we use non-uniform scaling?)
3. **The rendering function must be done from scratch, i.e. you must not use a third party library to draw your shape.**

4. In addition to the regular shape rendering, must be able to render the vertex and face normal. Refer following functionality requirements/guidelines for implementing this part:
   - Use GL_LINES to draw the normal. For rendering a face normal, place it at the centroid of the triangle. **(How would you compute a face normal for a triangle?)**
   - Compute the vertex normal by averaging the face normal of the faces that the vertex is a member of.
   - **Your program should provide a context menu-driven functionality to toggle between displaying the vertex normal, the face normal, or none (refer provided framework).**
   - Use suitable scale so that the normal vector does not appear too big wrt. the model.
5. Point 4 requires implementing a data structure that holds the geometry (per vertex information) and the topology (face membership information for all vertices). We will call this data structure a "triangle mesh."
6. **Rendering must be done using OpenGL rendering functions as discussed in class. Use Vertex Buffer Objects with the Index Buffer Objects for indexed mode rendering.**
7. The only "lighting" functionality that we will implement in this assignment is the ambient and diffuse lighting terms for two light sources. The details of this implementation are available in the shaders provided with the framework. We will expand this implementation in the second assignment.
8. **Hints:**
   - The geometric information in the OBJ file can be translated directly into suitable arrays for specifying vertices and their attributes.
   - The topological information includes the indices that will populate the so-called "index array" to be passed to the OpenGL rendering pipeline.
   - Depending on face-based or vertex-based normal specification, you may have to reallocate the vertex information to account for redundant shared vertices between

multiple faces. Your code MUST support this functionality for all types of rendering calls.

GUI functionality:

- Understand the basic GUI functionality provided and implement the following transformations using the GUI
  - Rotation using Euler angles (X, Y and Z)
  - Translation
- Populate the information under “Material properties” GUI section with assigned values that can be changed at run time.
- Modify Diffuse and Ambient terms in the “Light properties” GUI section. Implement this functionality with enough generality to include more lights in the future assignments. (In fact, you will include up to 8 lights simultaneously in the next assignment.) Currently, the GUI may not have a handler in the backend to change OpenGL parameters. You are expected to provide this functionality.

Assignment Submission Guideline

Please refer to the syllabus for assignment submission guideline. Failure to adhere to the submission guidelines might cause you to lose points.

Grade Sheet

Student Name: ______________________________ Points Total : ________________

Implementation Point	Grade	Points Obtained	Comments
OBJ File reader	**10%**		
File read correctly	5		
All entries parsed and identified correctly	5		
Normal Calculation and display	**20%**		
Vertex normals	10		
Face normals	10		
Rendering the geometry	**50%**		
Correct initialization and de-initialization of client state attributes	5		
All vertex attributes (position, normal) accounted for	5		
Correctly transferring vertex attributes to the GPU using indexed mode	10		
Indices used with correct offset	10		
Correct sequence of API calls to transfer data to GPU	10		
Rotation transforms implemented correctly	5		
Translation transform implemented correctly	5		
Specified GUI functionality (for the default lights)	**10%**		
Rollouts filled & updated with correct material data values	5		
Rollouts represent correct light attributes/fields	5		
Miscellaneous issues (code compilation, missing files)	**10%**		
Total	**100**		