E0 271: Computer Graphics

Assignment 1
Due: August 25, 2016
Weightage: 15%

Goals

- Learn OpenGL
- Learn basic GLSL
- Learn to use GLUT/Qt
- Analyze graphics performance

You need to implement a simple protein viewer in OpenGL and GLUT (or Qt). You will add more features to the protein viewer as a part of your next assignment. Note that equal points will be awarded for understanding the material. A basic OpenGL sample program is available at `http://vgl.csa.iisc.ac.in/e0271/sample.zip`. Built on top of this sample program and implement the following:

1. Load and display a protein surface mesh, available as an OFF file at `http://vgl.csa.iisc.ac.in/e0271/protein_surfaces.zip`. Determine and apply appropriate model transformations and orthographic projection to the protein mesh such that it appears in the middle of the screen. [25]
2. Allow rotation of the protein mesh about a fixed axis. Rotate the protein continuously at an appropriate rate. [15]
3. Measure the performance (in frames per second, and polygons rendered per second) of the system using a timer and display it on the window. [10]
4. Each protein surface is provided at 5 levels of detail, with `PDB_1.off` being the highest and `PDB_16.off` the lowest resolution mesh. Observe and report how the mesh resolution affects the performance. [10]
5. Color the vertices in the mesh based on their current depth (distance from the camera) using shaders. [10]
6. Slice the mesh in two halves using an axis parallel plane, apply appropriate transformations to the halves so that they are detached from each other, and display both halves. Allow the slicing plane to be changed on-the-fly. [30]

Notes:

- There are a number of ways to compute elapsed time. One is to use some method provided by your operating system. Another is to use glutGet(GLUT ELAPSED TIME) which returns the time in milliseconds since glutInit or the first invocation of the function. Most graphics cards allow the option of coupling the frame rate of the display and the redisplay of graphics. In Linux, this is sometimes done by setting an environment variable such as VSYNC TO BLANK. On Windows systems, you can often set this option through the display control panel. On the Mac, you can set it through a preferences menu that that you can pull down

when running an OpenGL program. For animated displays, you usually want this option on since most simple programs will generate their output in less than a frame time. However, for this assignment you probably do not want it on since the graphics card will most likely be sitting idle most of the time which will lead to misleading results.

- If you wish, you can use other OFF models instead of the provided protein surfaces. A few OFF models are available at `http://shape.cs.princeton.edu/benchmark/`.
- For task 6, you don't need to compute the exact geometric intersection of the protein mesh and the plane. That is, it is OK if you use the triangles in the original mesh to get an intersection boundary which is close to the true intersection boundary.

Turn in a copy of your code (with makefile, readme, etc.) and a text file containing the performance reports on or before date of submission between 4-6pm in VGL (Rm 237, CSA). Be prepared to demo your program during interactive grading. After finishing this assignment, you should have a grasp of OpenGL, GLSL and GLUT that you can build on for the next assignment. Be creative and have fun!

Additional tasks (not to be graded):

- **More transformations:** In addition to rotation about an axis, support interactive scaling, translation, and arbitrary rotation of the model.
- **Extension of task 6:** Use any arbitrary plane for slicing instead of an axis parallel plane; Compute the exact geometric intersection of the mesh with the plane; Use geometry shader to accomplish this task.
- **Deformation and animation:** Simulate twisting of the protein mesh. Hold the base of the protein mesh constant, and apply progressively higher rotation transformation to the vertices as you move towards the top of the mesh (Choose base and top planes appropriately). Animate this deformation till the top completes one full rotation.