COMPSCI 715 Problem Sheet 2, 2003.

A Bit of This and That

Due: Thursday 7 August, ***before*** *the lecture.*

All questions are equally weighted but are not *of equal difficulty. Do the easier stuff first!.*

1. Find a reference that explains Catmull-Clark subdivision (the bi-cubic sort, not the bi-quadratic version like that of Doo-Sabin). Then write a program that displays the mesh created by applying a single Catmull-Clark refinement step to a cube. You are not expected to implement the refinement in the program – you can compute the new vertices "by hand" first and just enter the definition of the polyhedron as data. Wire frame output is sufficient (though flat-shaded faces would be nicer ☺).
2. What is the relationship between the z-values that an OpenGL user specifies and the depths with which the depth buffer algorithm deals?
3. THIS QUESTION HAS BEEN DELETED! [It used to say … "Write a small OpenGL program that outputs the precision (i.e. number of bits) of each depth-buffer value. What answer do you get? Give the answer for your home machine if you have one, specifying what graphics card it has. Otherwise, give the answers for the lab machines."]
4. A terrain rendering program is using a near-plane distance of 1and a far-plane distance of 1000 (with assumed units of metres).
   (a) Whereabouts in the scene are depth-buffer errors most likely to occur? [By a "depth buffer error" I mean a failure of the algorithm to correctly determine that a pixel from one polygon is in front of a pixel from another polygon.]
   (b) What will happen when such errors occur?
   (c) In the worst case, what is the minimum necessary z-separation of two pixels in world-coordinates to guarantee no errors occur? Give an answer for both a 16-bit depth buffer and a 32-bit depth buffer.
5. Is it possible to set up a scene containing ONLY multiple non-intersecting axis-aligned cuboids (i.e. scaled cubes) that will defeat *any* depth-sort algorithm that does not clip scene polygons? Justify your answer.
6. A scene consists of a unit axis-aligned cube with one corner at (0,0,0) and the diagonally opposite corner at (1,1,1), sitting on the "ground", which is represented by a single large horizontal polygon. A BSP tree of the scene is built using the algorithm in the notes. The order of polygons in the list given to the initial constructor is: top, bottom, left, right, front, back, ground, where the cube face labels relate to a viewpoint along the positive z-axis (looking in the negative z direction).
   (a) Sketch the BSP tree created, making it clear exactly what the plane of each node is and what polygons or polygon fragments are present at each node.
   (b) What is the order of output polygons if the BSP tree is traversed with respect to a viewpoint at (5,5,0.5)?
7. As discussed in the notes, a BSP tree provides a depth ordering of polygons, but still results in all polygons being drawn to the screen. In very large complexity scenes (e.g. 100 million polygons), this is not acceptable. Discuss in some detail how BSP trees might be used to dramatically reduce the total number of polygons processed by the CPU and/or the graphics card.

Handing in and Marking

Write your answers neatly and *in ink* on A4 paper. Hand in your program code for questions 1 and 3 to the assignment drop box BEFORE the lecture. Bring your written answers plus relevant portions of program listings to the lecture.

This assignment is worth 3.5% of your grade for the course.