LSU EE 7700-2 Homework 3 Due: 31 March 2008

The solution template for Problem 1 of this assignment is available through SVN at https://svn.ece.lsu.edu/svn/gp/hw/2008/hw3 .

Problem 1: Modify the code in the solution template so that it uses triangle strips to specify the triangles making up the tube. (See the OpenGL documentation for use of triangle strips.) The homework template is written so that it will be easy to switch between triangle strips and independent triangles (once the triangle strip code is written). The triangle vertices using triangle strips (the added code) must be at the same coordinates as those with the individual triangles (the existing code). That is, when switching between the two (using the "t" key) there should be no change in the appearance of the tube.

- The code must support all three buffering methods (see the `opt_v_buffering` `switch` statement).
- It will probably be necessary to emit multiple strips, each for a ring of triangles.
- The code should continue to write all vertices in array `coor_buffer` before passing them to OpenGL, however vertices might be written in a different order when triangle strips are in use.
- For `opt_v_buffering` options 1 and 2 the entire `coor_buffer` and `norm_buffer` should be given to OpenGL, even if multiple triangle strips are needed. (See the documentation for `DrawArrays` in the OpenGL specification.)

Problem 2: Consider the performance benefit of using triangle strips in the previous problem.

(*a*) Show the change in performance between triangle strips and the original code for each of the three methods of vertex transfer.

(*b*) For each of the three methods of vertex buffering, either explain the observed change in performance or if you find the change in performance inexplicable, explain the change in performance that should have been obtained. If you haven't gotten the first problem working just explain what you thing the change in performance should be and why.

Problem 3: Using triangle strips will certainly reduce the amount of data passed to OpenGL API calls and it will certainly not reduce the amount of data written to the frame buffer.

(*a*) By what factor does the use of triangle strips reduce data passed to OpenGL API calls for the first problem? (An answer of 0.5 indicates that half the data is transferred?) (This part is easy.)

(*b*) Consider a GPU and OpenGL implementation in which the amount of data passed from the CPU to the GPU is the same whether or not triangle strips are used.

How might that be the fault of the OpenGL implementation (say, if the the OpenGL implementation was grudgingly developed by a company with its own proprietary GPU API)?

How might that be due to the design of the GPU? Assume that the GPU is moderately well designed but either old or targeted at a particular part of the market. Be reasonably specific on why a GPU might make it impossible to reduce the amount of data passed.

Problem 4: Consider the GPU designs presented in class and the use of triangle strips from the first problem. The use of triangle strips will certainly reduce the amount of data passed to OpenGL and certainly **not** reduce the data written to the frame buffer. How far down the GPU hardware (rendering pipeline) can the reduction be seen? That is, at what point will the reduction disappear?