COSC422 Advanced Computer Graphics

Programming Exercise 05

Surface of Revolution

The geometry shader stage is a very powerful and versatile stage in the OpenGL pipeline. It provides access to whole primitives entering the pipeline and is useful in generating new vertices and primitives. In this exercise, we will use a geometry shader to output a surface of revolution. The application generates only a simple two-dimensional line strip representing the base curve for the surface.

The file `Vase.dat` contains the 2D coordinates of 19 vertices of a curve segment. The program `GS_SurfRevln.cpp` generates a display of this curve as a line strip (Fig. 1)

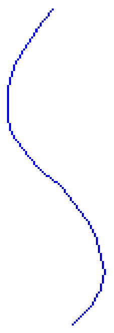

Fig. 1

Create a geometry shader as shown on COSC363 Lecture Slide [11]-17 to access each line segment of the input strip. Convert the vertex shader to a pass-through shader by removing the matrix multiplication operation. The geometry shader will now receive the end points of each line segment in world coordinates in the built-in variables `gl_in[].glPosition`. Using the *x*-coordinates of the input points as radii, we build a triangle strip as shown below (Fig. 2):

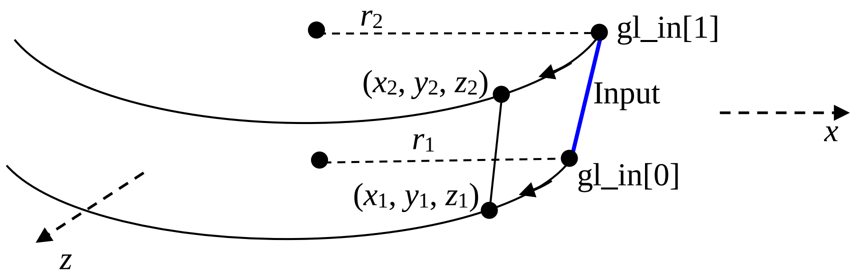

Fig. 2.

Inside the geometry shader, generate 36 pairs of points (at 10 deg intervals) along the circular ring with coordinates of the points (Fig. 2) given by (r_1 cosθ, y_1 , r_1 sinθ), and (r_2 cosθ, y_2 , r_2 sinθ) , where,

y_1 = gl_in[0].gl_Position.y

y_2 = gl_in[1].gl_Position.y

Each generated point is converted to clip coordinates and output using the built-in out variable gl_Position. The function EmitVertex() must be called when a geometry shader outputs a vertex.

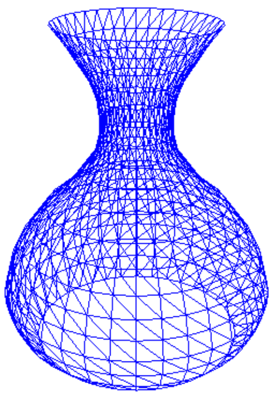
Fig. 3.

[11]: COSC363 Lecture slides, “Geometry Shader”.