

# **Geometry Shader**

**CSCI 4239/5239**

**Advanced Computer Graphics**

**Spring 2016**

# What is it?

- Create new primitives
  - Point → Polygon
- Inserted between vertex shader and fragment shader
- Changes each glVertex into multiple vertexes

# OpenGL Implementation

- Create and compile just like others
  - `glCreateShader(GL_GEOMETRY_SHADER)`
- Requires additional parameters
  - In the program
    - `glProgramParameteri(prog,par,val);`
    - `GL_GEOMETRY_INPUT_TYPE`
    - `GL_GEOMETRY_OUTPUT_TYPE`
    - `GL_GEOMETRY_VERTICES_OUT`
  - In the shader
    - `layout(type) in;`
    - `layout(type,max_vertices=n) out;`
- Append EXT on older versions of OpenGL

# GLSL Implementation

- Set vertex parameters like in vertex shader
  - `gl_FrontColor`
  - `gl_TexCoord`
  - `gl_Position`
- Call `EmitVertex()`; when done
- Call `EndPrimitive()`; after last vertex

# Application: n-Body Problem

- Movement of  $n$  bodies under gravitational influence
- Classical problem in computational dynamics
- Hard because effort grows as  $n^2$
- Display locations of bodies

# Digression: OpenMP

- Multi-threaded approach
  - Lightweight
  - Needs shared memory
- API supported in C/C++ using pragmas

```
#pragma omp parallel for
for (k=0;k<N;k++)
    foo(k);
```
- Simple to use
- Needs compiler support
  - gcc -fopenmp

# Ex 20: OpenMP+Geometry Shader

- Solve n-Body problem using OpenMP
  - Euler integration
  - Ping-Pong implementation
- Use geometry shader to turn points into a quad and billboard
  - Apply texture to point
  - Blend to add
- Example of a particle shader

