Compute Shaders

CSCI 4239/5239
Advanced Computer Graphics
Spring 2018

Purpose of Compute Shaders

- Lightweight general purpose computing
  - Perform arbitrary computations outside of the vertex/fragment paradigm
  - Access to textures and buffers
  - Does not require additional drivers or run time
  - Easier initialization and invocation than OpenCL
- Requires OpenGL 4.3
  - Example 22 requires GL 4.4

Using the Compute Shader

- Compiles just like other shaders
  - CreateShader(prog,GL_COMPUTE_SHADER,file)
  - Link only one shader into program
- Bind and access buffers
  - glBindBuffer(GL_SHADER_STORAGE_BUFFER,x)
  - glBufferData()
    - Set size and type of buffer
  - glMapBufferRange()
    - Access to buffer on CPU
  - glUnmapBuffer(GL_SHADER_STORAGE_BUFFER);

Running the Compute Shader

- glUseProgram(compute_shader)
  - Select program
- glDispatchCompute(Nx,Ny,Nz)
  - Run Nx,Ny,Nz groups
- glDispatchComputeGroupSize(Nx,Ny,Nz,x,y,z)
  - Set both number of groups and work groups
- glMemoryBarrier(GL_SHADER_STORAGE_BARRIER_BIT)
  - Wait for compute shaders

Pre-set Variables in Shader

- uvec3 gl_NumWorkGroups
- uvec3 gl_WorkGroupSize
- uvec3 gl WorkGroupID
- uvec3 gl_LocalInvocationID
- uvec3 gl_GlobalInvocationID
- uvec3 gl_LocalInvocationIndex

- gl_GlobalInvocationID =
  gl_WorkGroupID *gl_WorkGroupSize + gl_LocalInvocationID
- gl_LocalInvocationIndex =
  gl_LocalInvocationID.z*gl_WorkGroupSize.y *gl_WorkGroupSize.x +
  gl_LocalInvocationID.y * gl_WorkGroupSize.x +
  gl LocalInvocationID.x