

GPU and GPGPU Programming

Shuai Lu 170742

Assignment 1

Task

1. Setup

Install Visual Studio 2015 or higher or choose other suitable IDE for C++.

Install CUDA from <https://developer.nvidia.com/cuda-downloads>

Install CMake.

From the folder of the assignment (where the CMakeLists.txt file is located) run the command 'cmake .' to generate a project.

Compile and run

2. OpenGL Query and print (to console):

OpenGL version and available extensions:

GL vendor, renderer, and version

Find out how to query extensions with GLEW (<http://glew.sourceforge.net/>, <http://www.opengl.org/registry/>).

Query and print the extensions your GPU supports.

Query and print GPU OpenGL limits:

maximum number of vertex shader attributes

maximum number of varying floats

number of texture image units (in vertex shader and in fragment shader, respectively)

maximum 2D texture size

maximum number of draw buffers

other information of your choice

3. CUDA Query and print CUDA functionality:

number of CUDA-capable GPUs in your system using `cudaGetDeviceCount()`

CUDA device properties for every GPU found using `cudaGetDeviceProperties()`:

device name

compute capability: driver version, runtime version, major and minor rev. numbers

multi-processor count

clock rate

total global memory

shared memory per block

num registers per block

warp size (in threads)

max threads per block

4. GLFW Read up on GLFW. Try to implement the callbacks for mouse interaction and keyboard events. Simply print to the console that you have detected a mouse move/click or a keyboard event.

5. Commit and push your solution and a short report that includes the output of your program

Outcome:

This assignment is finished on Linux environment. The related repositories and software are installed successfully.

The OpenGL vendor, renderer, version and extensions are queried and printed as shown in Fig.1.

GPU OpenGL limits are queried and printed in Fig.2.

CUDA functionality is queried using function `cudaGetDeviceProperties()`. The information is printed as follow:

The window is created by using GLFW framework. The position of the cursor, the behaviors of the mouse and the keyboard are traced by using function `glfwSetCursorPosCallback()`, `glfwSetMouseButtonCallback()`, and `glfwSetKeyCallback()`. The related callback functions are also created. The detected information is printed on the console(Fig.4).

```
lus0a@lus0a: ~/CS380/cs380-2021/1_assignment
[100%] Built target assignment1
lus0a@lus0a:~/CS380/cs380-2021/1_assignment$ ./assignment1
----- OpenGL version and extensions -----
GL Vendor      : NVIDIA Corporation
GL Renderer    : Quadro K2200/PCIe/SSE2
GL Version     : 4.6.0 NVIDIA 470.57.02
GL_AMD_multi_draw_indirect
GL_AMD_seamless_cubemap_per_texture
GL_ARB_arrays_of_arrays
GL_ARB_base_instance
GL_ARB_bindless_texture
GL_ARB_blend_func_extended
GL_ARB_buffer_storage
GL_ARB_cl_event
GL_ARB_clear_buffer_object
GL_ARB_clear_texture
GL_ARB_clip_control
GL_ARB_color_buffer_float
GL_ARB_compatibility
GL_ARB_compressed_texture_pixel_storage
GL_ARB_conservative_depth
GL_ARB_compute_shader
GL_ARB_compute_variable_group_size
GL_ARB_conditional_render_inverted
GL_ARB_copy_buffer
GL_ARB_copy_image
GL_ARB_cull_distance
GL_ARB_debug_output
GL_ARB_depth_buffer_float
GL_ARB_depth_clamp
GL_ARB_depth_texture
GL_ARB_derivative_control
GL_ARB_direct_state_access
GL_ARB_draw_buffers
GL_ARB_draw_buffers_blend
GL_ARB_draw_indirect
GL_ARB_draw_elements_base_vertex
GL_ARB_draw_instanced
GL_ARB_enhanced_layouts
GL_ARB_ES2_compatibility
```

Figure 1: OpenGL query and print

```
----- GPU OpenGL limits -----
Maximum number of vertex shader attributes      : 16
Maximum number of varying floats : 124
Number of texture image units in fragment shader : 32
Number of texture image units in vertex shader : 32
Maximum 2D texture size : 16384
Maximum 3D texture size : 4096
Maximum number of draw buffers : 8
```

Figure 2: GPU OpenGL limits

```
----- CUDA functionality -----
Device(s): 1
Device Name:  Quadro K2200
Compute capability:  5.0
Multi-processor count:  5
Clock rate:  1124000
Total global memory:  4234870784
Shared memory per block:  49152
Num registers per block:  65536
Warp size in threads:  32
Max threads per block:  1024
```

Figure 3: CUDA functionality

```
Left button is clicked and cursor location is (194.000000,266.000000)
Right button is clicked and cursor location is (308.000000,215.000000)
Left button is clicked and cursor location is (144.000000,256.000000)
Key ESC is pressed, the window is closed
```

Figure 4: The record of mouse and keyboard events