Reading Assignment #1 (until Sep 2)

Read (required):

- Orange book, chapter 1 (*Review of OpenGL Basics*)
- Orange book, chapter 2 (*Basics*)
Syllabus (1)

GPU Basics and Architecture (~August, September)

- Introduction
- GPU architecture
- How shader cores work
- GPU shading and GPU compute APIs
  - General concepts and overview
  - Learn syntax details on your own!
    - GLSL book
    - CUDA book
    - Online resources, ...
Syllabus (2)

GPUs for Graphics (~October)

- GPU texturing, filtering
- GPU (texture) memory management
- GPU frame buffers
- Virtual texturing
Syllabus (3)

GPU Computing (~November)
- GPGPU, important parallel programming concepts
- CUDA memory access
- Reduction, scan
- Linear algebra on GPUs
- Deep learning on GPUs
- Combining graphics and compute
  - Display the results of computations
  - Interactive systems (fluid flow, ...)

Semester project presentations
Programming Assignments: Basics

5 assignments

• Based on C/C++, OpenGL, and CUDA

Organization

1. Explanation in readme, and during lecture (and Q&A sessions if required)
2. Get framework online (bitbucket+git)
3. Submit solution and report online (bitbucket+git) by submission deadline
4. Personal presentation after submission
Programming Assignments: People

Teaching Assistants:

• Peter Rautek (*peter.rautek@kaust.edu.sa*) – programming assignments; assignment presentations
  
  Office: Bldg 1, Room 2220
Programming Assignments: People

- CS380 machines
- Markus Hadwiger office #2219
- Peter Rautek office #2220

Visual Computing Center Building 1
1. Google, Stackoverflow, …

2. Ask your fellow students

3. Contact us: Peter Rautek peter.rautek@kaust.edu.sa
Playing with the GPU

GPU programming comes in different flavors:

- Graphics: OpenGL, DirectX, Vulkan
- Compute: CUDA, OpenCL

In this course we will:

- Learn to use CUDA and OpenGL
- Wrap our heads around parallelism
- Learn the differences and commonalities of graphics and compute programming

Format:

- 5 Pre-specified programming assignments
- 1 Capstone (semester) project that you can define yourself
• Source code is hosted on bitbucket.org
• Register with your kaust.edu.sa email address (will give you unlimited plan – nice!)
• Go to the repo https://bitbucket.org/rautek/cs380-2019 (or simply search on bitbucket for cs380) and fork it
• Get a git client http://git-scm.com/downloads and clone your own repo
• Follow the readme text-file
• Do your changes in the source code for assignment 1, commit, and push (to your own repo)
• Contact Peter Rautek if you have problems or questions (peter.rautek@kaust.edu.sa)
Programming Assignment 1

Set up your development environment

- Visual Studio 2015 (or 2017, 2019)
  (https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=Community&rel=16)
- git (https://git-scm.com/downloads)
- Fork the CS380 repository (https://bitbucket.org/rautek/cs380-2019)
- Follow the readme and start coding

Query your graphics card for its capabilities (CUDA and OpenGL)
Programming Assignment 1 – Setup

• Programming
  • Query hardware capabilities (OpenGL and CUDA)
  • Instructions in readme.txt file
• Submission (via bitbucket)
  • Program
    • Short report (1-2 pages, pdf), including short explanation of program, problems and solutions, how to run it, screenshots, etc.
• Personal assessment
  • Meeting at Peter’s office (building 1, office 2220)
  • Max. 15 minutes, present program and source code
Programming Assignments: Grading

- Submission complete, code working for all the required features
- Documentation complete (report, but also source code comments!)
- Personal presentation
- Optional features, coding style, clean solution
- Every day of late submission reduces points by 10%
- No direct copies from the Internet!
  You have to understand what you program:
  your explanations during the presentations will be part of the grade!
Assignment #1:
  • Querying the GPU (OpenGL/GLSL and CUDA) due Sep 4

Assignment #2:
  • Phong shading and procedural texturing (GLSL) due Sep 18

Assignment #3:
  • Image Processing with GLSL due Oct 2

Assignment #4:
  • Image Processing with CUDA
  • Convolutional layers with CUDA due Oct 16

Assignment #5:
  • Linear Algebra (CUDA) due Nov 6
Semester Project

- Choosing your own topic encouraged!
  (we will also suggest some topics)
  - Pick something that you think is really cool!
  - Can be completely graphics or completely computation, or both combined
  - Can be built on CS380 frameworks, NVIDIA OpenGL SDK, or CUDA SDK
- Write short (1-2 pages) project proposal by end of Sep (announced later)
  - Talk to us before you start writing!
    (content and complexity should fit the lecture)
- Submit semester project with report (deadline: Dec 5)
- Present semester project (we will schedule event in final exams week)
What are GPUs?

**Graphics Processing Units**

But evolved toward

- Very flexible, massively parallel floating point co-processors
- But not entirely programmable!
- Fixed-function parts have definite advantages (e.g., texture filtering, z-buffering)

We will cover both perspectives

- GPUs for graphics
- GPU computing (GPGPU – general purpose computation on GPU)
What is in a GPU?

Lots of floating point processing power

- Stream processing cores
  - different names: stream processors, CUDA cores, ...
  - Was vector processing, now scalar cores!

Still lots of fixed graphics functionality

- Attribute interpolation (per-vertex -> per-fragment)
- Rasterization (turning triangles into fragments/pixels)
- Texture sampling and filtering
- Depth buffering (per-pixel visibility)
- Blending/compositing (semi-transparent geometry, ...)
- Frame buffers
Peak Performance

Theoretical GFLOP/s

- NVIDIA GPU Single Precision
- NVIDIA GPU Double Precision
- Intel CPU Double Precision
- Intel CPU Single Precision

Geforce 780 Ti
Geforce GTX Titan
Geforce GTX 680
Geforce GTX 580
Geforce GTX 480
Geforce GTX 280
Geforce 8800 GTX
GeForce 7800 GTX
GeForce 6800 Ultra
Woodcrest
Harpertown
Bloomfield
Westmere
Ivy Bridge
Sandy Bridge
Tesla M2090
Tesla K20X
Tesla K40
Pentium 4
Apr-01
Sep-02
Jan-04
May-05
Oct-06
Feb-08
Jul-09
Nov-10
Apr-12
Aug-13
Dec-14
Peak Performance

Theoretical GFLOP/s at base clock

- NVIDIA GPU Single Precision
- NVIDIA GPU Double Precision
- Intel CPU Single Precision
- Intel CPU Double Precision

Markus Hadwiger, KAUST
Peak Bandwidth
RISE OF GPU COMPUTING

- APPLICATIONS
- ALGORITHMS
- SYSTEMS
- CUDA
- ARCHITECTURE

Graph showing the growth of GPU and CPU performance from 1980 to 2020, with a projection to 2025. The GPU performance is projected to grow at a rate of 1.5X per year, while the CPU performance grows at a rate of 1.1X per year.

GPU Architectures Over the Years

GPU Roadmap

- Volta (Stacked DRAM)
- Maxwell (Unified Virtual Memory)
- Kepler (Dynamic Parallelism)
- Fermi (FP64)
- Tesla (CUDA)

after Volta:
- Turing (2018/2019)
- Ampere (2020)
Real-time graphics primitives (entities)

Represent surface as a 3D triangle mesh

Vertices

Primitives (e.g., triangles, points, lines)

Courtesy Kayvon Fatahalian, CMU
Real-time graphics primitives (entities)

- Vertices
  - 1
  - 2
  - 3
  - 4

- Primitives
  - (e.g., triangles, points, lines)

- Fragments

- Pixels (in an image)

Courtesy Kayvon Fatahalian, CMU
Thank you.