

CS 380 - GPU and GPGPU Programming

Lecture 1: Introduction

Markus Hadwiger, KAUST



Lecture Overview

Goals

- Learn GPU architecture and programming; both for graphics and for compute (GPGPU)
- Shading languages (**GLSL**, HLSL, MSL, Cg), compute APIs (**CUDA**, OpenCL, DirectCompute)

Time and location

- Sunday + Wednesday, 14:30 – 16:00, Room 3128, Bldg. 9

Webpage:

https://vccvisualization.org/CS380_GPU_and_GPGPU_Programming/

Contact

- **Markus Hadwiger:** markus.hadwiger@kaust.edu.sa
- **Peter Rautek** (main contact assignments): peter.rautek@kaust.edu.sa
- **Amani Ageeli** (programming questions): amani.ageeli@kaust.edu.sa

Prerequisites

- **C/C++ programming (!), basic computer graphics, basic linear algebra**



Lecture Structure

Lectures

- Part 1: GPU Basics and Architecture (both: graphics, compute)
- Part 2: GPUs for Graphics
- Part 3: GPUs for Compute

Some lectures might be on research papers (both seminal and current)

Assignments

- 5 programming assignments
- Weekly reading assignments (required; also some optional)

Quizzes

- 4 quizzes, throughout the semester, 30 min each; announced at least a week in advance
- From lectures and (required) reading assignments

Semester project + final presentations, but no mid-term/final exam!

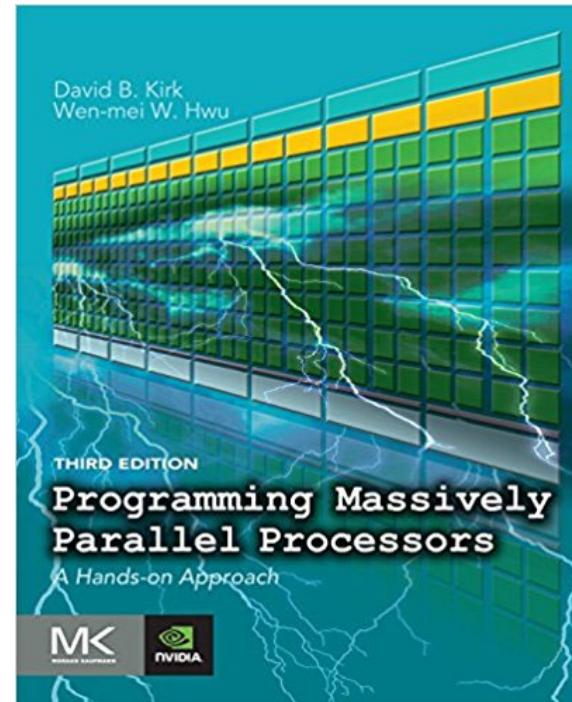
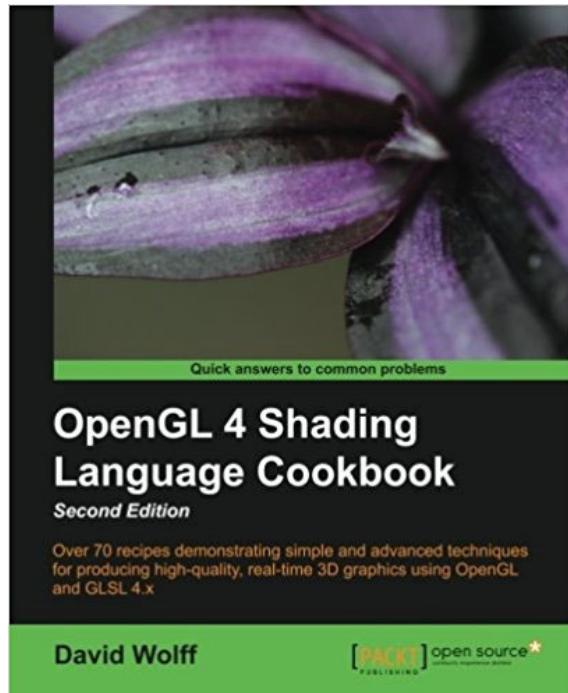
Grading: 40% programming assignments; 30% semester project; 30% quizzes



Resources (1)

Textbooks

- GPUs for Graphics: OpenGL 4 Shading Language Cookbook, 2nd or 3rd ed.
- GPU Computing / GPGPU: Programming Massively Parallel Processors, 3rd ed.



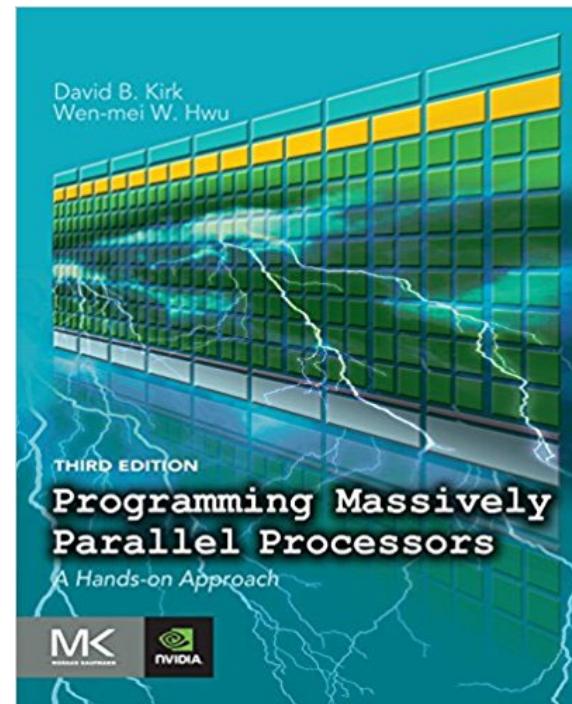
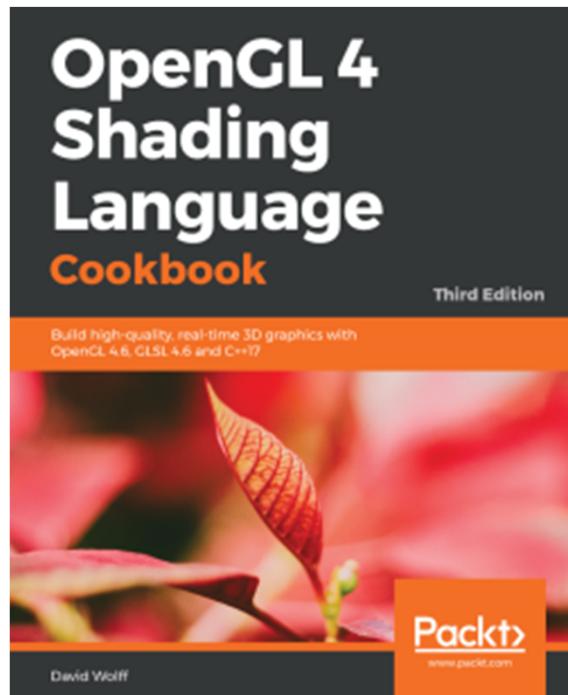
3rd ed.



Resources (1)

Textbooks

- GPUs for Graphics: OpenGL 4 Shading Language Cookbook, 2nd or 3rd ed.
- GPU Computing / GPGPU: Programming Massively Parallel Processors, 3rd ed.



3rd ed.



Resources (2)

https://vccvisualization.org/CS380_GPU_and_GPGPU_Programming/

- OpenGL (4.6): www.opengl.org
www.khronos.org/files/opengl46-quick-reference-card.pdf
- CUDA (11.7): developer.nvidia.com/cuda-toolkit/
- Vulkan (1.3): www.vulkan.org
- OpenCL (3.0): www.khronos.org/opencl/

Very nice resources for examples:

- *GPU Gems* books 1-3 (available online)
- *GPU Computing Gems*, Vol. 1 + 2 (Emerald/Jade edition)
- *Ray Tracing Gems* (2019) and *Ray Tracing Gems II* (2021)



Resources (3)

Learn OpenGL

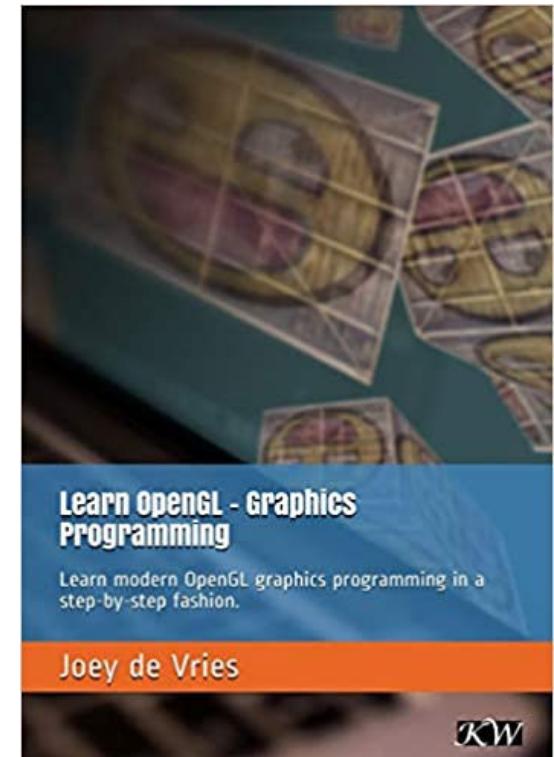
Nice recent introduction to OpenGL

Webpage:

<https://learnopengl.com/>

Free book as pdf:

https://learnopengl.com/book/book_pdf.pdf





Resources (4)

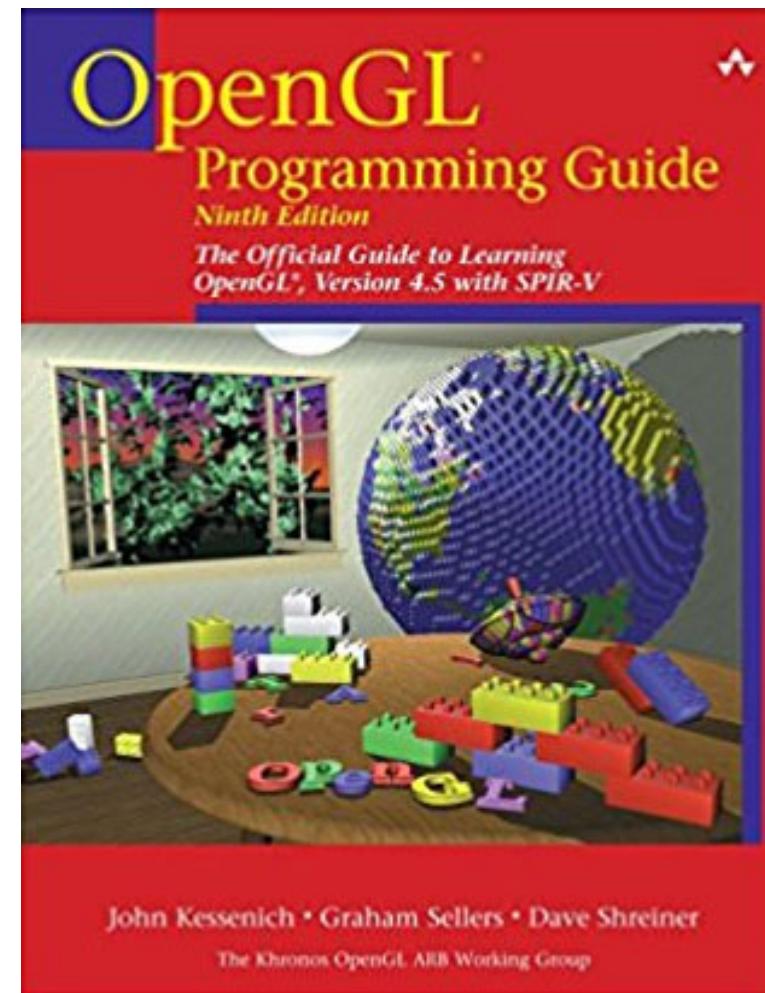
OpenGL Programming Guide (red book)

<http://www.opengl-redbook.com/>

Computer graphics and OpenGL

Current edition: 9th
OpenGL 4.5 (with SPIR-V)
contains extended chapters on GLSL

Available in the KAUST library
also electronically

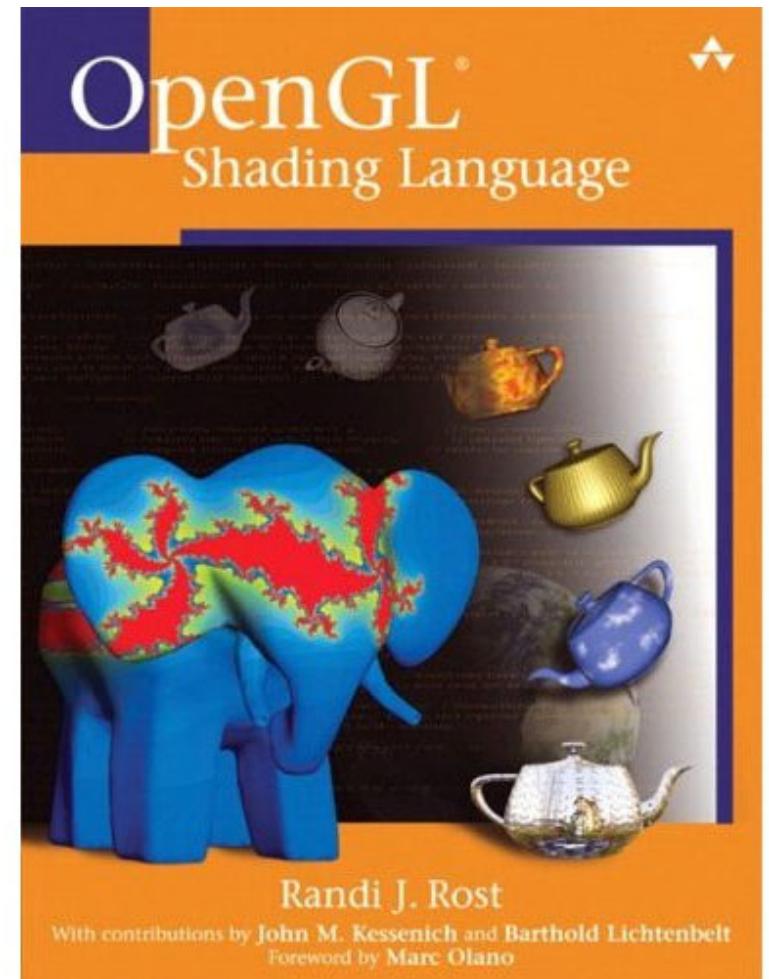


Resources (5)



OpenGL Shading Language (orange book)

Current edition: 3rd
OpenGL 3.1, GLSL 1.4
no geometry shaders
(outdated in several aspects,
but the basics are still very nice!)



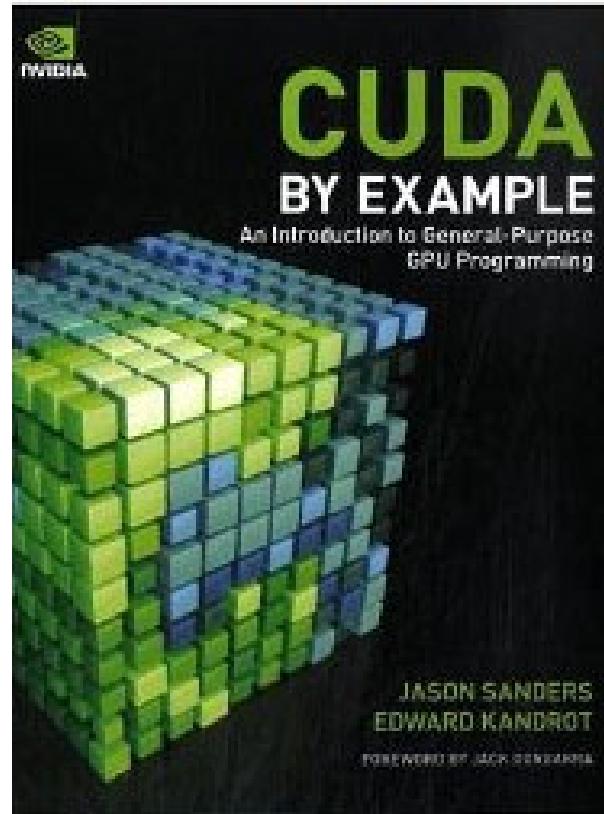
Available in the KAUST library
also electronically



Resources (6)

CUDA by Example: An Introduction to General-Purpose GPU Programming, Jason Sanders, Edward Kandrot

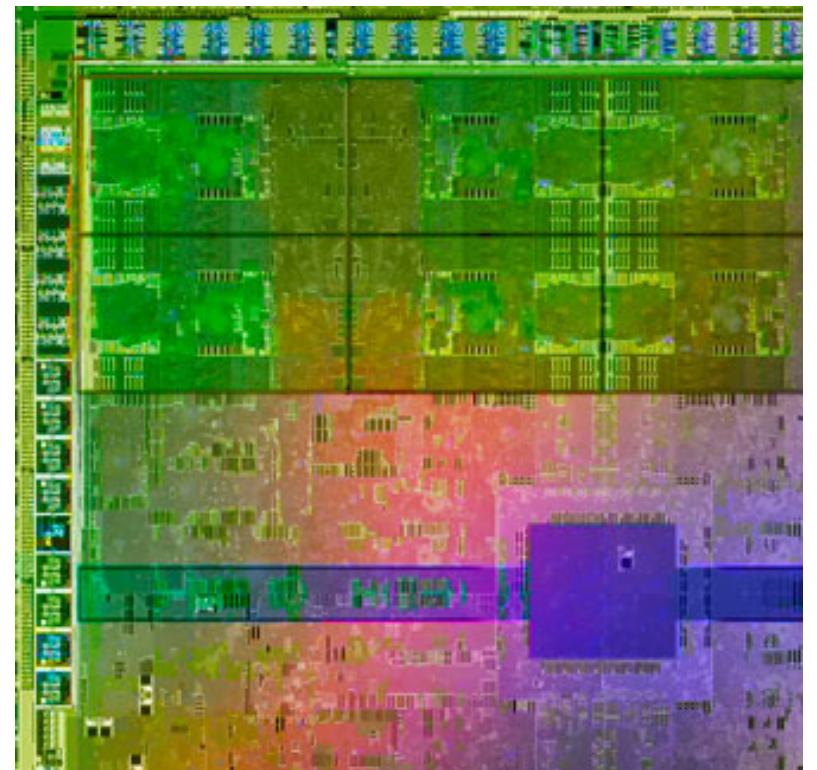
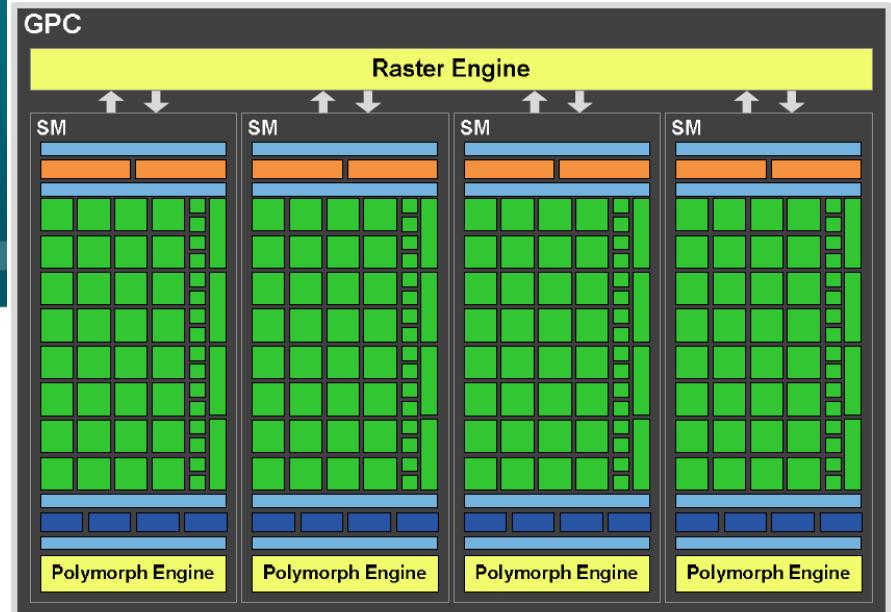
See reference section
of KAUST library



Syllabus (1)

GPU Basics and Architecture (~August, September)

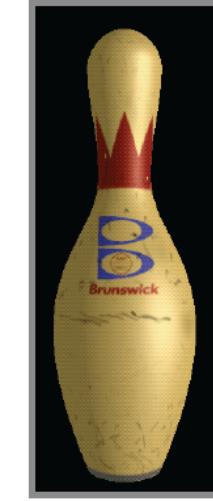
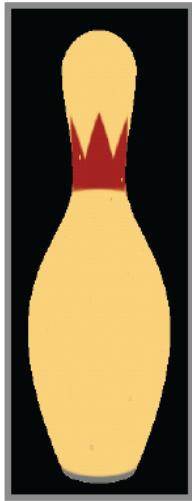
- Introduction
- **GPU architecture**
- How shader cores work
- GPU shading and GPU compute APIs
 - General concepts and overview
 - Learn syntax details on your own !
 - GLSL book
 - CUDA book
 - Online resources, ...



Syllabus (2)

GPUs for Graphics (~October)

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

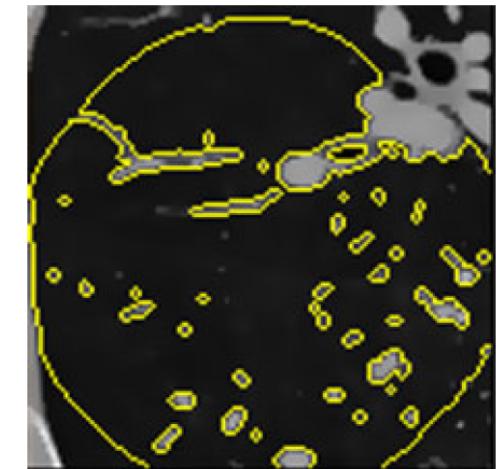


Syllabus (3)



GPU Computing (~November, December)

- 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, ...)



segmentation

SAXPY
 $\begin{array}{c} \square \square \square \square \square \\ + \\ \alpha \cdot \square \square \square \square \end{array}$

SGEMV
 $\begin{array}{c} \square \\ . \\ \square \square \end{array}$

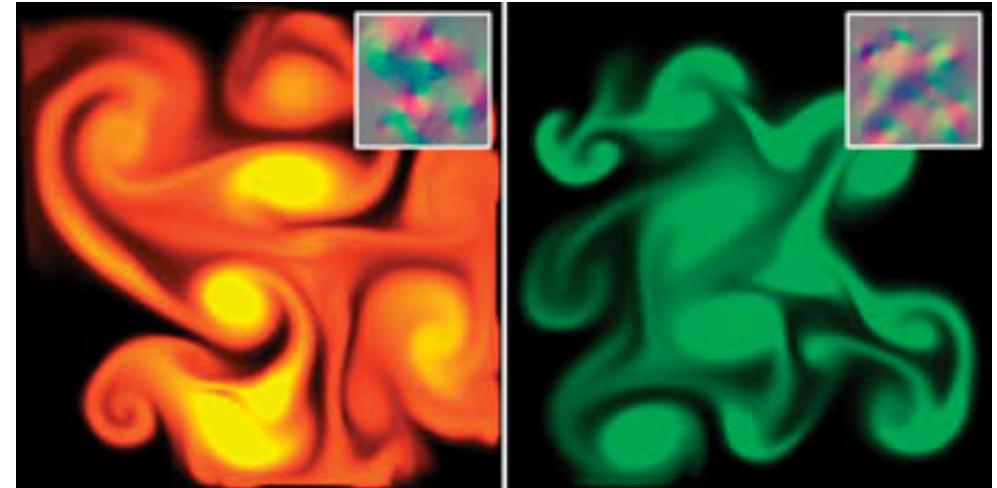
Semester project presentations

linear algebra

Example: Fluid Simulation and Rendering



- Compute advection of fluid
 - (Incompressible) Navier-Stokes solvers
 - Lattice Boltzmann Method (LBM)
- Discretized domain; stored in 2D/3D textures
 - Velocity, pressure
 - Dye, smoke density, vorticity, ...
- Updates in multi-passes
- Render current frame

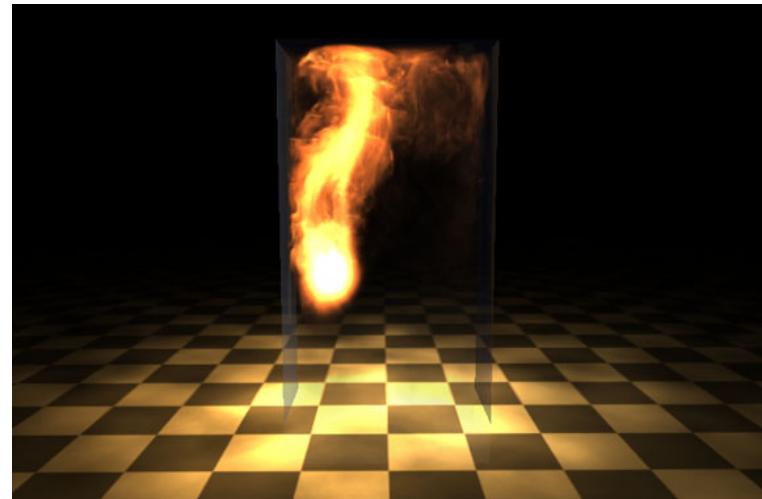


Courtesy Mark Harris



Example: Volumetric Special Effects

- NVIDIA Demos
 - Smoke, water
 - Collision detection with voxelized solid (Gargoyle)
- Ray-casting
 - Smoke: direct volume rendering
 - Water: level set / isosurface



Courtesy Keenan Crane 15



Example: Ray Tracing

Ray tracing in hardware (ray tracing cores: ray/triangle intersect, BVH)

- Microsoft DXR (DX12 Ultimate API), Vulkan, NVIDIA OptiX
- NVIDIA Turing: “World’s First Ray Tracing GPU” Quadro RTX, GeForce RTX
- AMD RDNA 2 (also in PS5, Xbox Series X), upcoming Intel Arc (Alchemist, 2022)



Epic Games Unreal Engine 4 with MS DXR

Example: Particle Simulation and Rendering



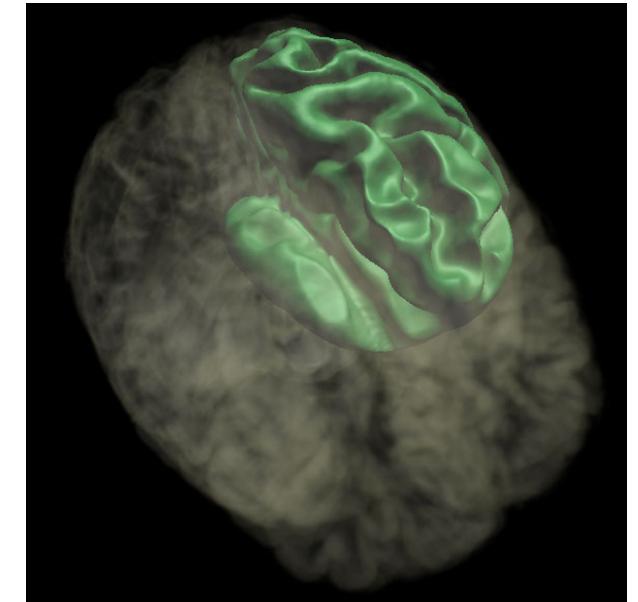
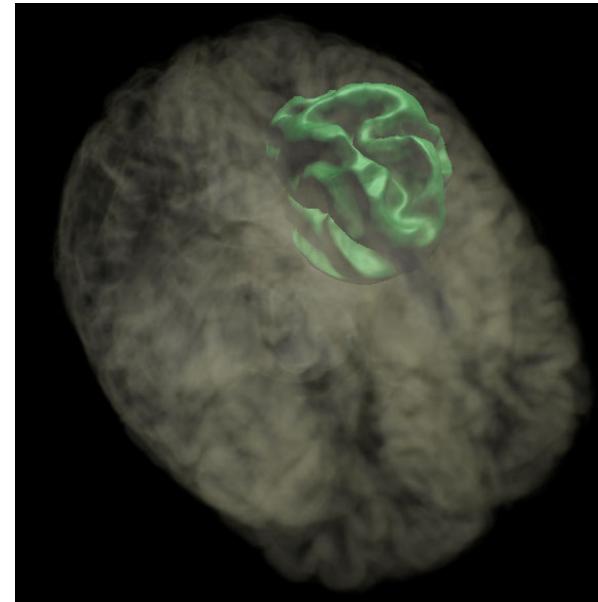
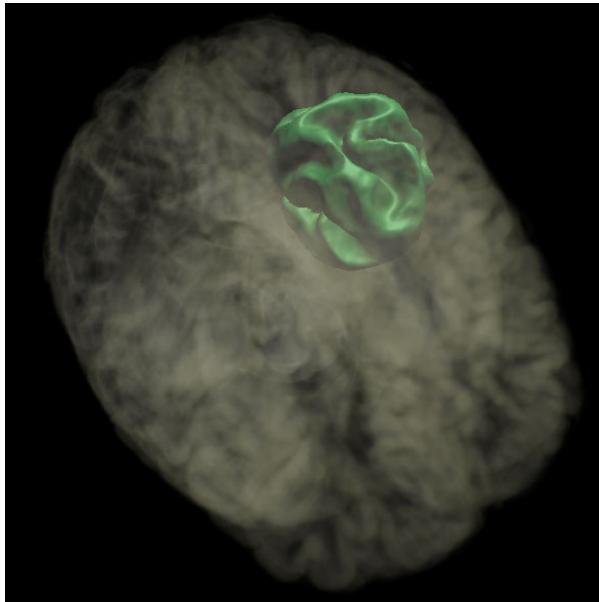
- NVIDIA Particle Demo





Example: Level-Set Computations

- Implicit surface represented by distance field
- The level-set PDE is solved to update the distance field
- Basic framework with a variety of applications

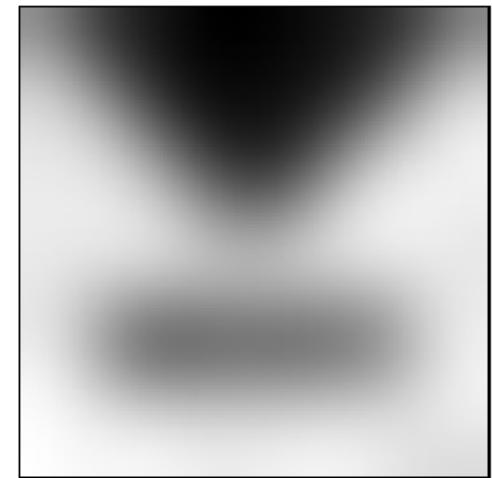
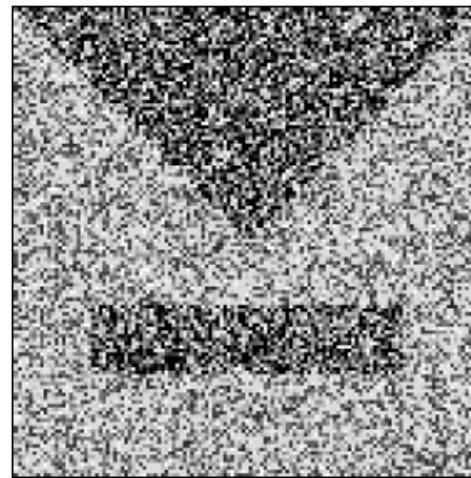




Example: Diffusion Filtering

De-noising

- Original
- Linear isotropic
- Non-linear isotropic
- Non-linear anisotropic

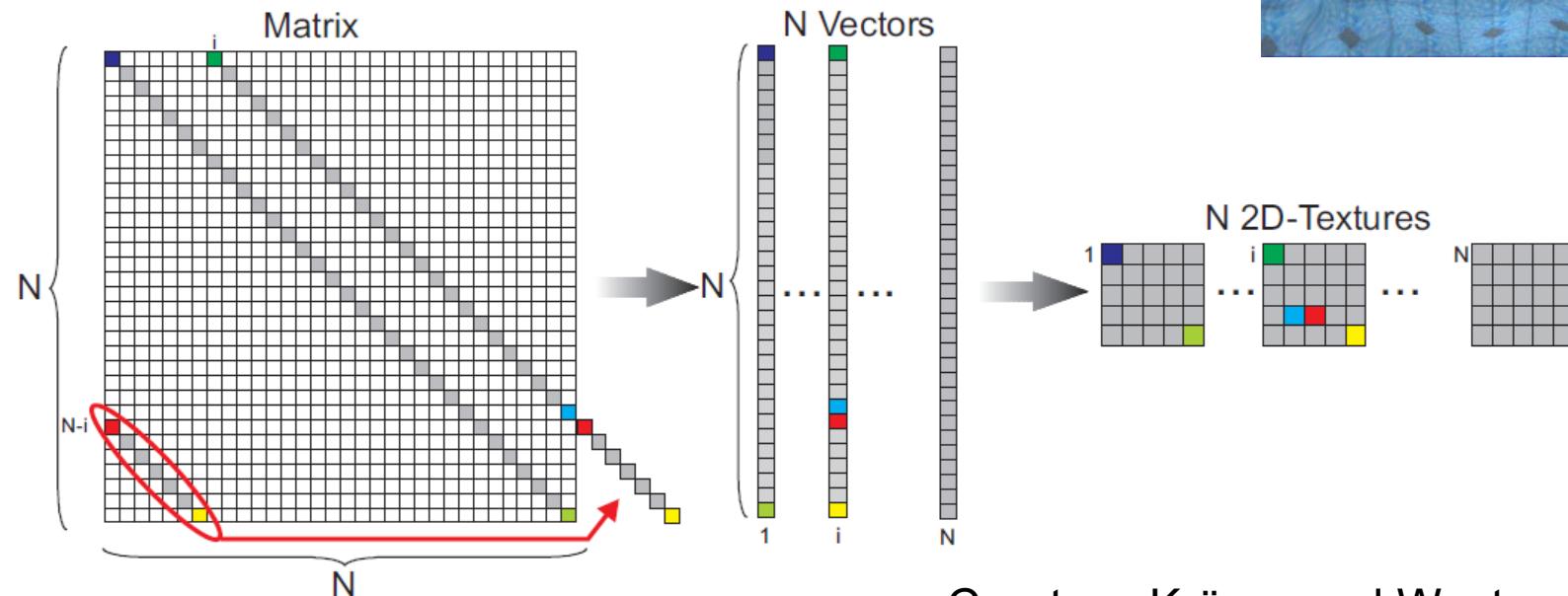
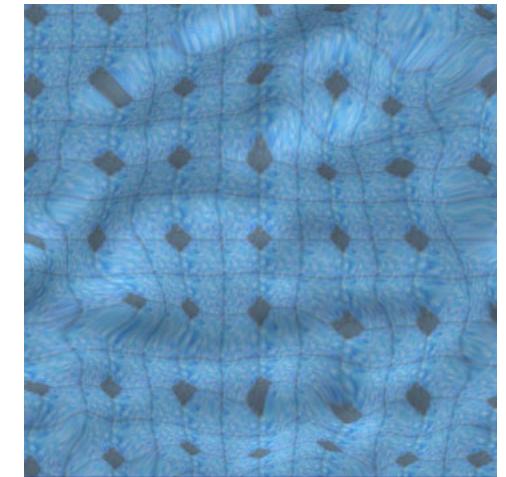




Example: Linear Algebra Operators

Vector and matrix representation and operators

- Early approach based on graphics primitives
- Now CUDA makes this much easier (+ lots of libraries)
- Linear systems solvers



Courtesy Krüger and Westermann
20



Example: Machine Learning / Deep Learning

Perfect fit for massively parallel computation

- NVIDIA Volta Architecture: Tensor Cores (mixed-prec. 4x4 matrix mult plus add)
- NVIDIA Turing and Ampere architectures: Improved tensor cores, ...

Frameworks

- TensorFlow,
PyTorch,
Caffe,
- ...

WHY ARE GPUs GOOD FOR DEEP LEARNING?

	Neural Networks	GPUs
Inherently Parallel	✓	✓
Matrix Operations	✓	✓
FLOPS	✓	✓
Bandwidth	✓	✓

GPUs deliver --

- same or **better** prediction accuracy
- faster results
- smaller footprint
- lower power
- lower cost

Higher layer (Model V3?)

Higher layer (Model V2?)

Model V1

Input image

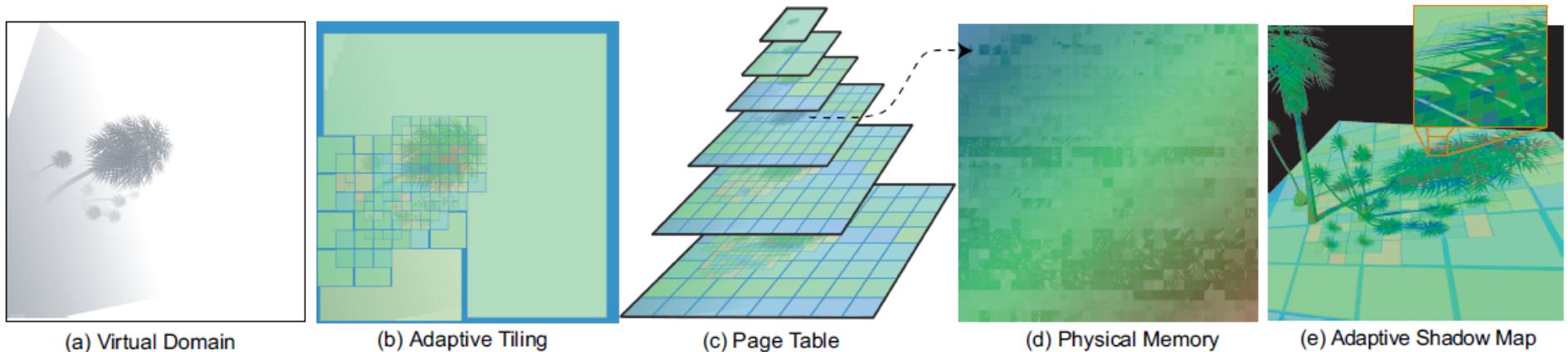
[Lee, Ranganath & Ng, 2007]

Example: GPU Data Structures



Glift: Generic, Efficient, Random-Access GPU Data Structures

- “STL“ for GPUs
- Virtual memory management



Courtesy Lefohn et al.

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
- Amani Ageeli (amani.ageeli@kaust.edu.sa) – programming questions; general help





Need Help?

1. Google, Stackoverflow, ...
2. Ask your fellow students!
Discussions and explanations are encouraged!
(but: copying code is not allowed)
3. Contact us:
Peter peter.rautek@kaust.edu.sa
Amani amani.ageeli@kaust.edu.sa



Playing with the GPU

GPU programming comes in different flavors:

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

In this course we will:

- Learn to use CUDA and OpenGL (you can use other APIs for semester project!)
- 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

Programming Assignments: Where to Start



- 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/cs380-2022/src/main/>
(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)

OpenGL Tutorial



One extra session (attendance optional, but highly recommended!)

To make it easier to get started with OpenGL

Amani will do the tutorial with you

If you already have some questions / problems when you come to the tutorial, that's even better!

Doodle link to find a date+time that is suitable for everyone:

<https://doodle.com/meeting/participate/id/dRo01kVe>

Programming Assignment 1



Set up your development environment

- Visual Studio 2015, 2017, 2019, 2022
(<https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=Community&rel=16>)
 - CUDA 11.7 (<https://developer.nvidia.com/cuda-downloads>)
 - git (<https://git-scm.com/downloads>)
 - Fork the CS 380 repository (<https://bitbucket.org/rautek/cs380-2022/src/main/>)
 - 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
 - (Zoom) meeting with Peter
 - Max. 15 minutes, present program + source code

```
\\10.68.74.73\10_gpgpu\CS380_2012_Assignment_1_Solution\CS380_2012_Assignment_1\bin\Release\OpenGL Check Driver Supports and Information
--> OpenGL Check
GL Vendor : NVIDIA Corporation
GL Renderer : Quadro 6000/PCI/SSE2
GL Version : 4.1.0
GLEW Version : 1.7.0
3D Texture : Supported
3D Texture Array : Supported
2D Texture Array : Supported
2D Texture Size : 16384
2D Depth Size : 2048
Framebuffer Objects : Supported
Max Draw Buffers : 8
Max Tex Units Vert : 32
Max Tex Units Geom : 32
Max Tex Units Frag : 32
Max Vertex Attributes : 16
Max Varying Floats : 64
GLSL Version : Supported
GLSL Geom Shader <ARB> : Supported
GLSL Geom Shader <EXT> : Supported

--> CudaCheck There are 2 devices supporting CUDA
--> Device 1 Quadro 6000
CUDA Capability : 2.0
CUDA MP Count : 1
CUDA Cores : 448
Global Memory : 4.000 GB
Shared Memory : 48.000 KB
Registers / Block : 32768
Clock rate GPU : 1.147 GHz
Clock rate Memory : 1.494 GHz
Warp Size : 32
CUDA Threads / Block : 1024
CUDA Threads / Block : 1024 x 1024 x 64
CUDA Blocks / Grid : 65535 x 65535 x 65535
2D Texture Size : 65536 x 65535
3D Texture Size : 2048 x 2048 x 2048
CUDA Timeout : true

--> Device 2 Quadro 6000
CUDA Capability : 2.0
CUDA MP Count : 1
CUDA Cores : 448
Global Memory : 4.000 GB
Shared Memory : 48.000 KB
Registers / Block : 32768
Clock rate GPU : 1.147 GHz
Clock rate Memory : 1.494 GHz
Warp Size : 32
CUDA Threads / Block : 1024
CUDA Threads / Block : 1024 x 1024 x 64
CUDA Blocks / Grid : 65535 x 65535 x 65535
2D Texture Size : 65536 x 65535
3D Texture Size : 2048 x 2048 x 2048
CUDA Timeout : true

--> CudaCheck Driver Supports and Information
CUDA Driver Version : 4.0
CUDA Driver Version : 4.0
```



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!

Programming Assignments: Schedule (tentative)



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:

- Deferred Shading and Image Processing with GLSL due Oct 2

Assignment #4:

- Image Processing with CUDA
- Convolutional layers with CUDA due Oct 23

Assignment #5:

- Linear Algebra (CUDA) due Nov 13

Semester / Capstone 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 CS 380 frameworks, NVIDIA OpenGL SDK, 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 8)**
- Present semester project, event in final exams week: Dec 12 (tentative!)



Reading Assignment #1 (until Sep 4)

Read (required):

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

Thank you.