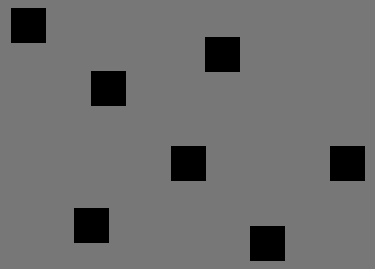


GPU Computing & CUDA

Han Xiao

What is GPU?

A Graphics Processing Unit (GPU) is a single-chip processor primarily used to manage and boost the performance of video and graphics.

- 2-D or 3-D graphics,
- Digital output to flat panel display monitors,
- Texture mapping,
- Rendering polygons,
- Hardware overlays,
- MPEG decoding

.....

These features are designed to lessen the work of the CPU and produce faster video and graphics

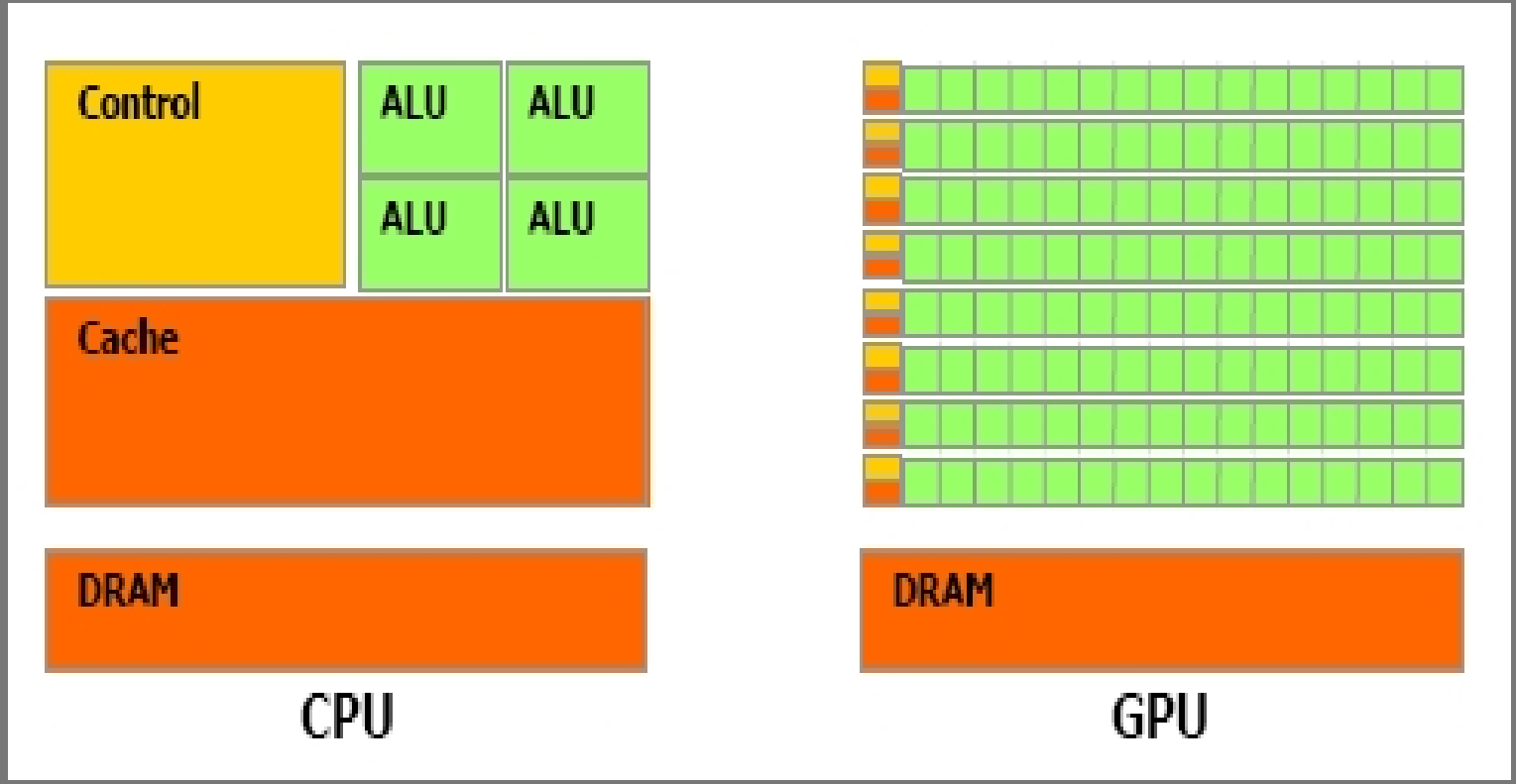
Contents

What is GPU Computing?

GPU Computing is the use of a GPU to do general purpose scientific and engineering computing.

- CPU and GPU together in a heterogeneous computing model
- Sequential part of the application runs on the CPU and the computationally-intensive part runs on the GPU
- From the user's perspective, the application just runs faster because it is using the high-performance of the GPU to enhance performance

Why GPU Computing?



Why GPU Computing?

Over the past few years, the GPU has evolved from a **fixed-function special-purpose** processor into a **full-fledged parallel programmable** processor with additional fixed-function special-purpose functionality

- Computational requirements are large
- Parallelism is substantial
- Throughput is more important than latency

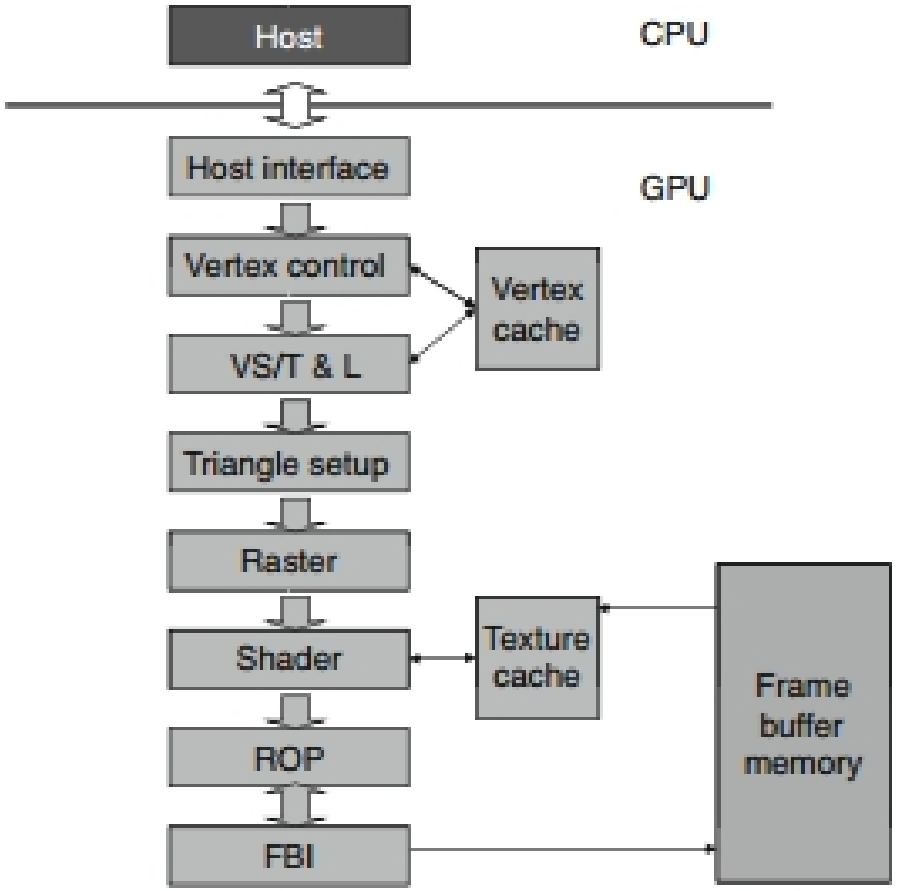
GPU Architecture

WHAT IS Graphics Pipeline?

The original GPUs were modeled after the concept of a graphics pipeline. The graphics pipeline is conceptual model of stages that graphics data is sent through, and is usually implemented via a combination of hardware(GPU cores) and CPU software(OpenGL, DirectX).

GPU Architecture

The Graphics Pipeline



Hints: !!

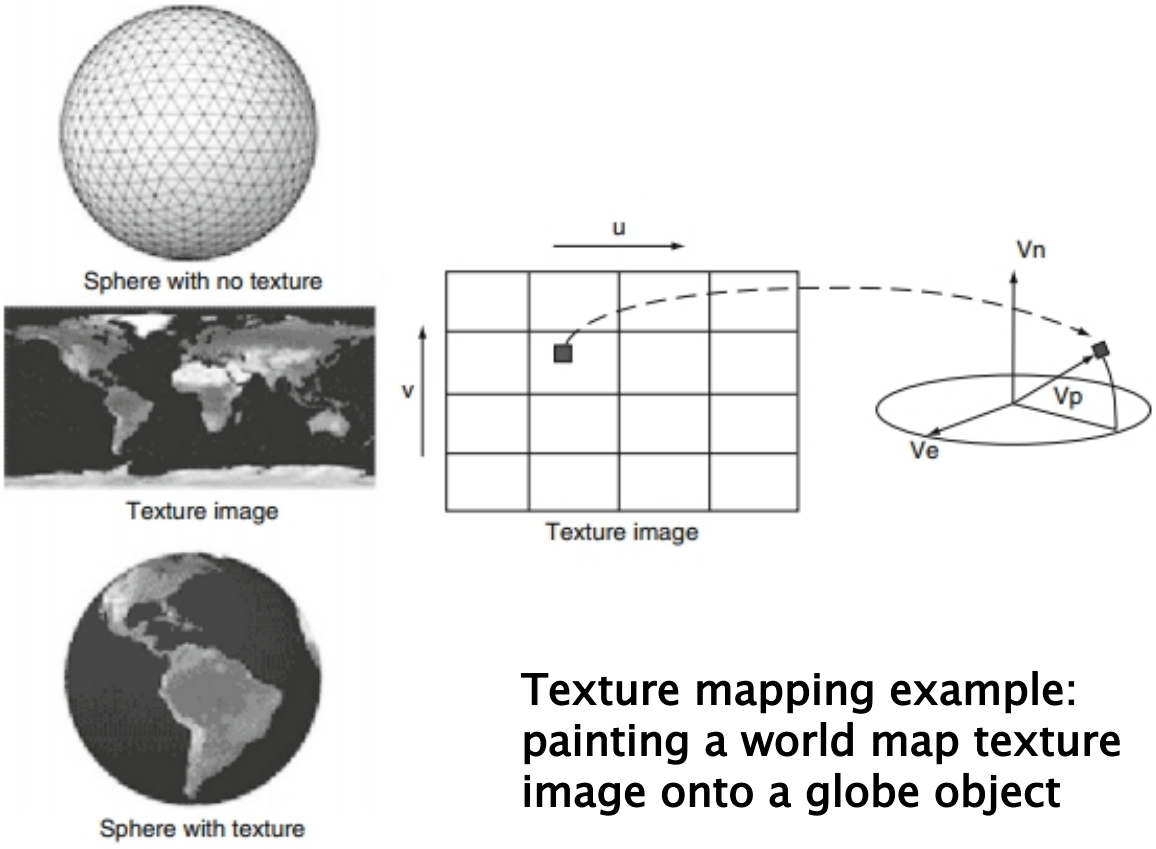
VS/T&L: Vertex shading, transform, and lighting

ROP: raster operation

FBI: frame buffer interface

GPU Architecture

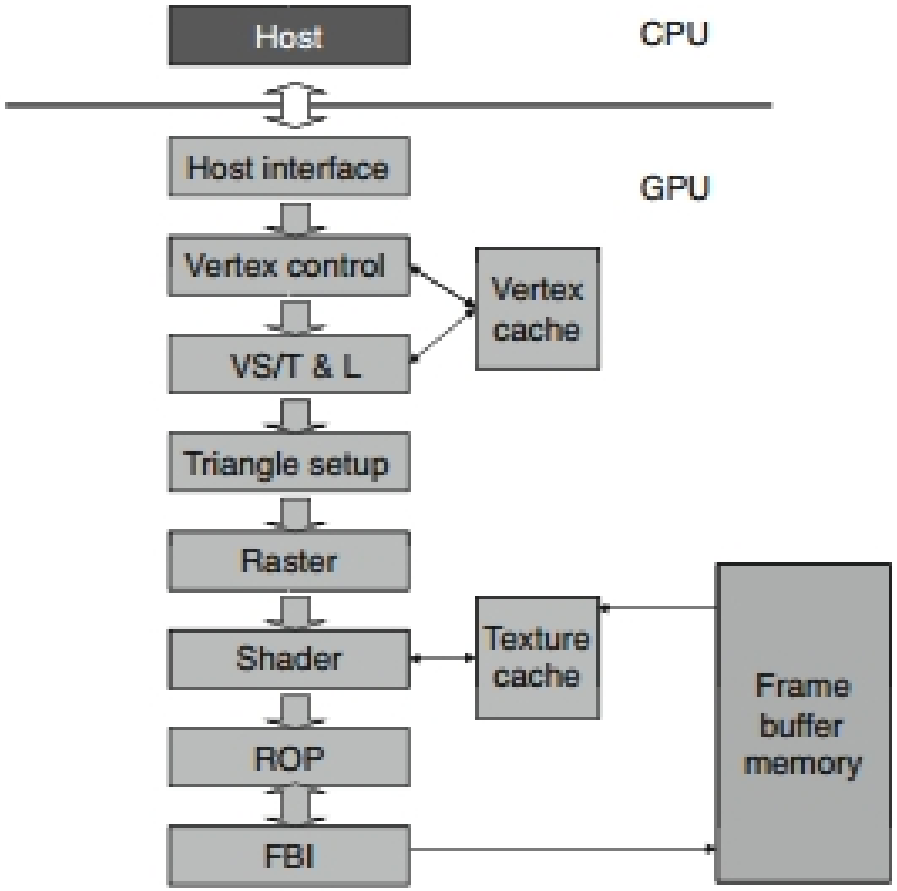
The Graphics Pipeline



Texture mapping example: painting a world map texture image onto a globe object

GPU Architecture

The Graphics Pipeline



Hints: !!

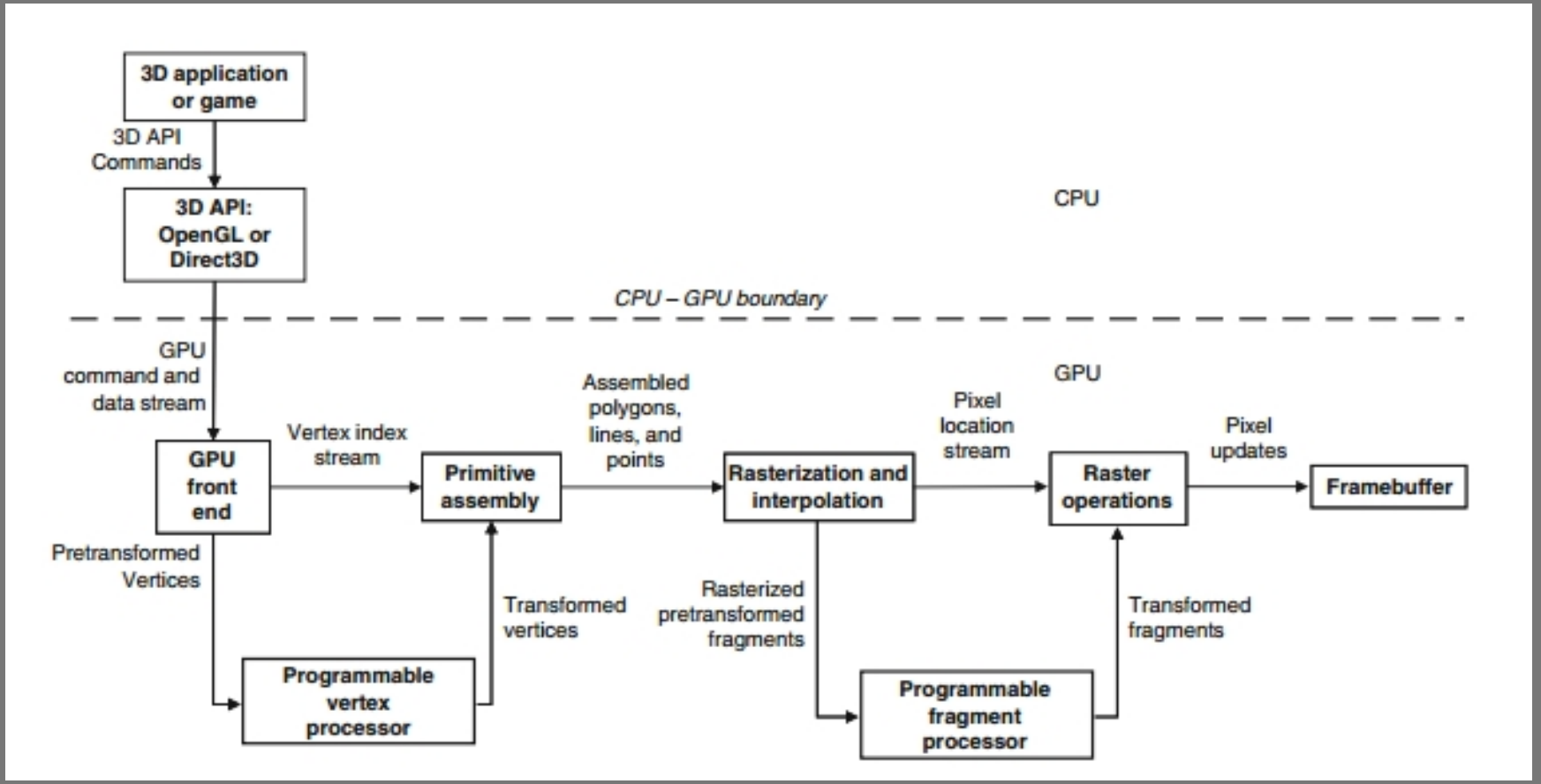
VS/T&L: Vertex shading, transform, and lighting

ROP: raster operation

FBI: frame buffer interface

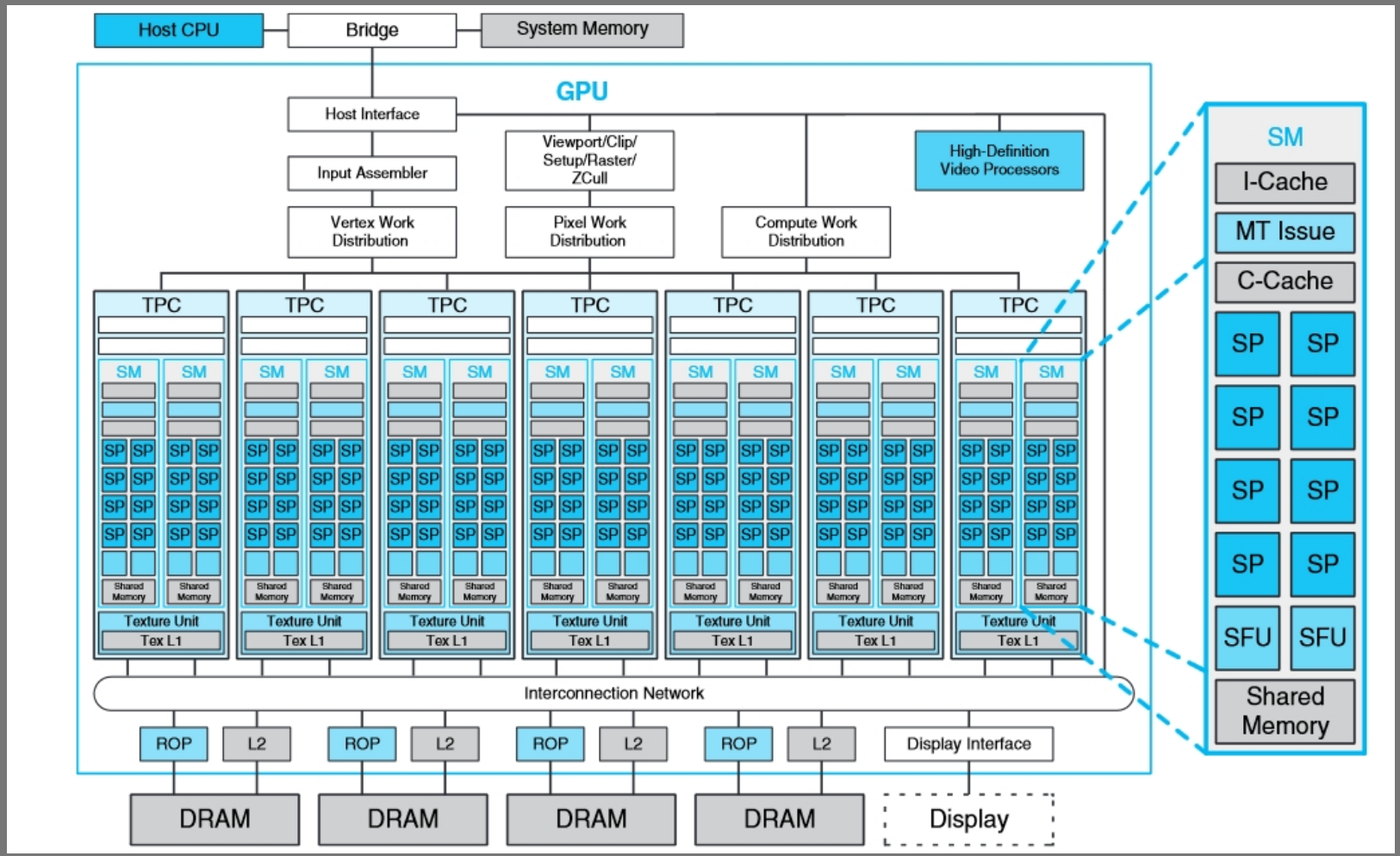
GPU Architecture

Programmable Real-Time Graphics



GPU Architecture

Unified Graphics and Computing Processors

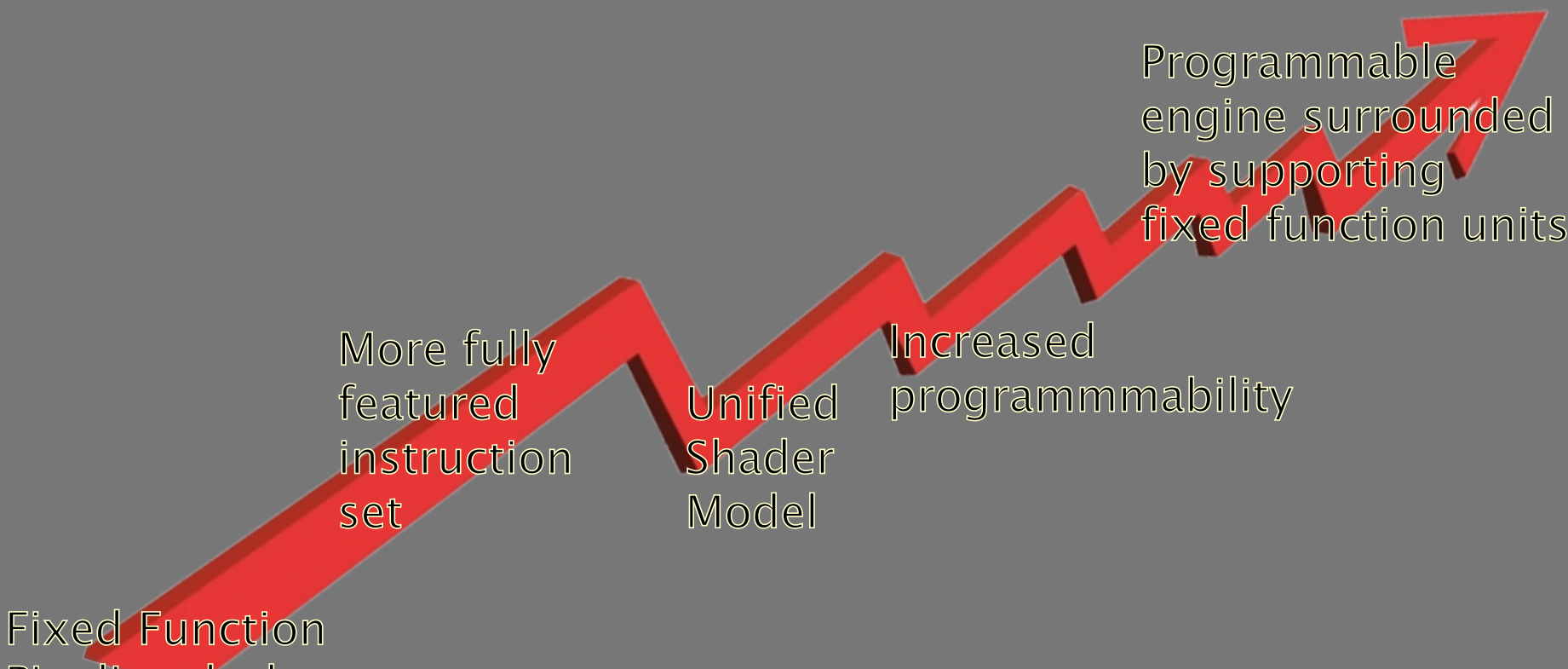


Further Reading:

- Akeley, K. (1993). Reality engine graphics.Computer Graphics
- Akeley, K., & Jermoluk, T. (1988). High-performance polygon rendering.
- Blythe, D. (2006). The Direct3D 10 System.ACM Transactions on Graphics

GPU Architecture

Evolution of GPU Architecture



GPU Architecture

Evolution of GPU Architecture

CUDA Model

What is CUDA?????

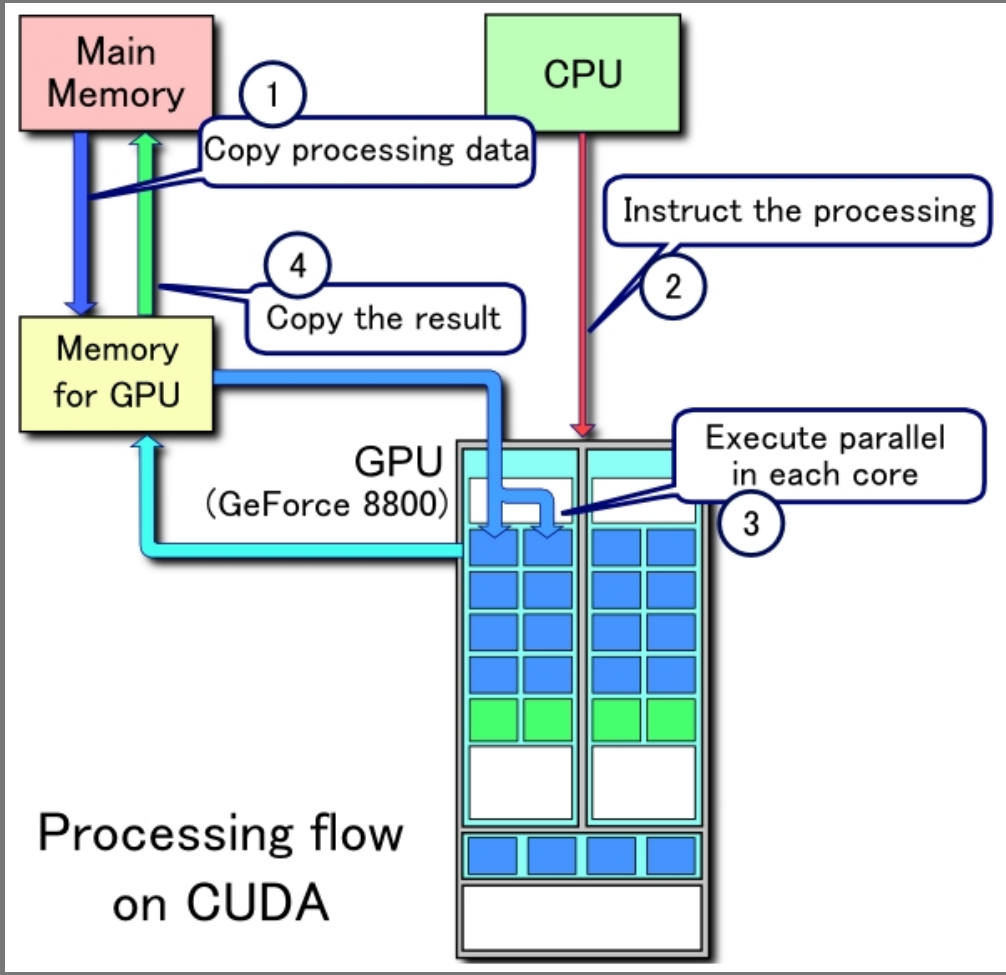
"Compute Unified Device Architecture"

A scalable parallel programming model and language based on C/C++

Is a parallel programming platform for GPUs and multicore GPUs

CUDA Model

CUDA processing flow



CUDA Model

CUDA Kernels

Parallel portion of application: execute as a kernel
- Entire GPU executes kernel
- Kernel launches create thousands of CUDA threads efficiently

CPU	Host	Executes functions
GPU	Device	Executes kernels

Kernel lauches create hierarchical groups of threads
- Threads are grouped into blocks, and blocks into grids
- Threads and blocks represent different levels of parallelism

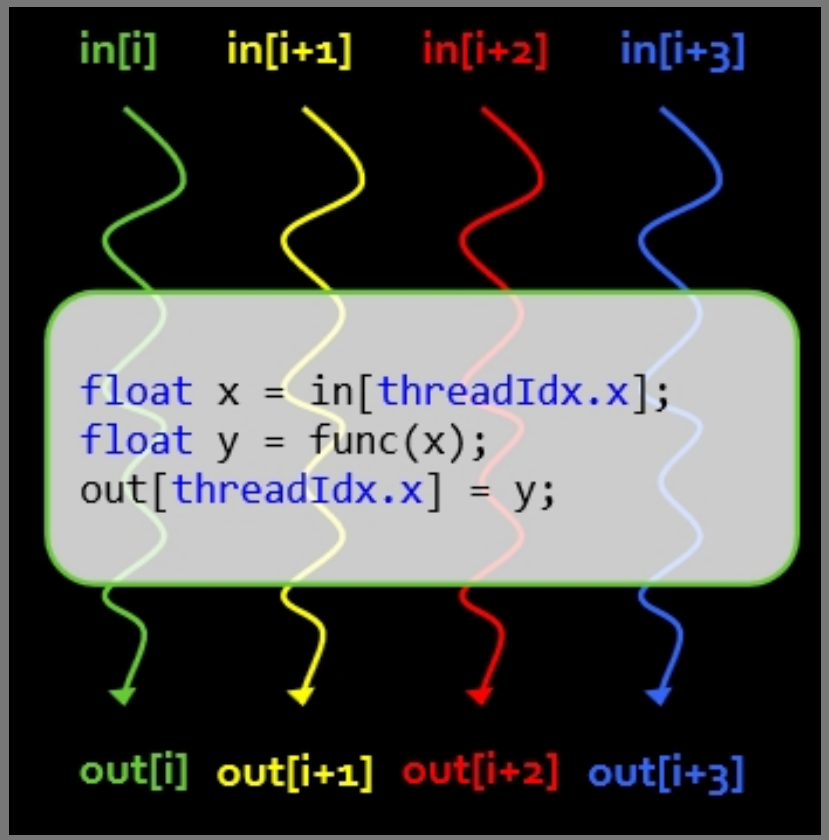
CUDA Model

CUDA Kernels: Parallel Threads

A kernel is a function executed on the GPU as an array of parallel threads

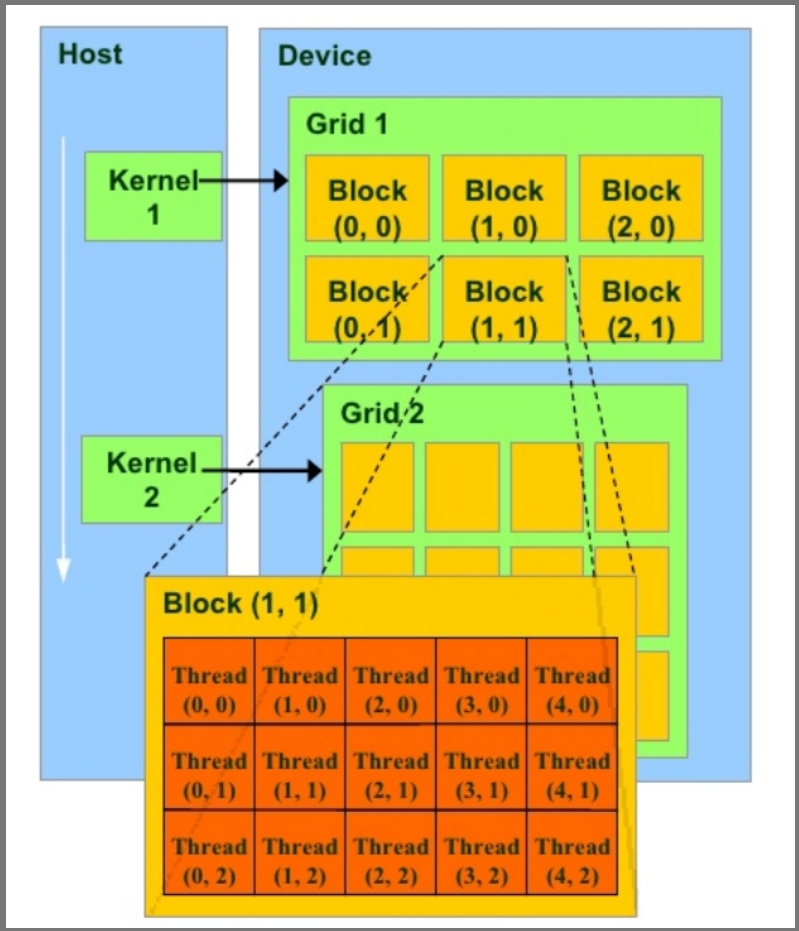
All threads execute the same kernel code, but can take different paths

Each thread has an ID
- Select input/output
- Control decisions



CUDA Model

Thread Organization



Thread, Block, Dimension

Thread
3D IDs, unique within a block

Block
2D IDs, unique within a grid

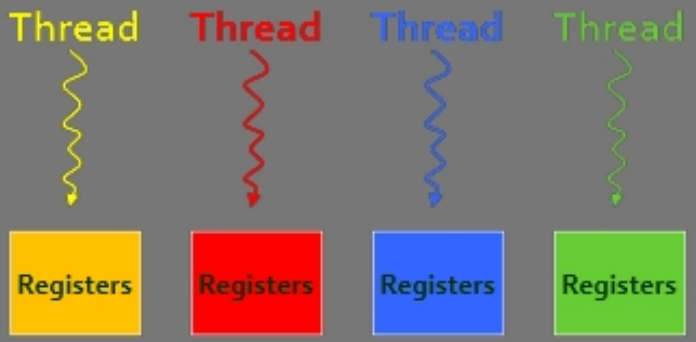
Dimension
can be unique for each gird

Built in variables
threadIdx, blockIdx
blockDim, gridDim

CUDA Model

CUDA Memory

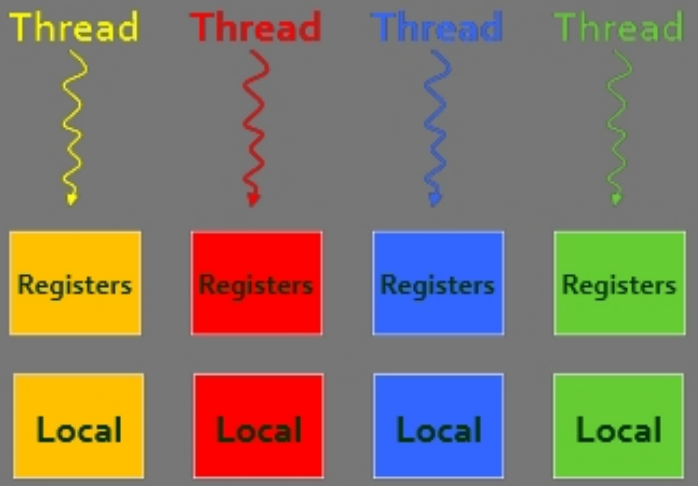
- Thread
  - Registers



CUDA Model

CUDA Memory

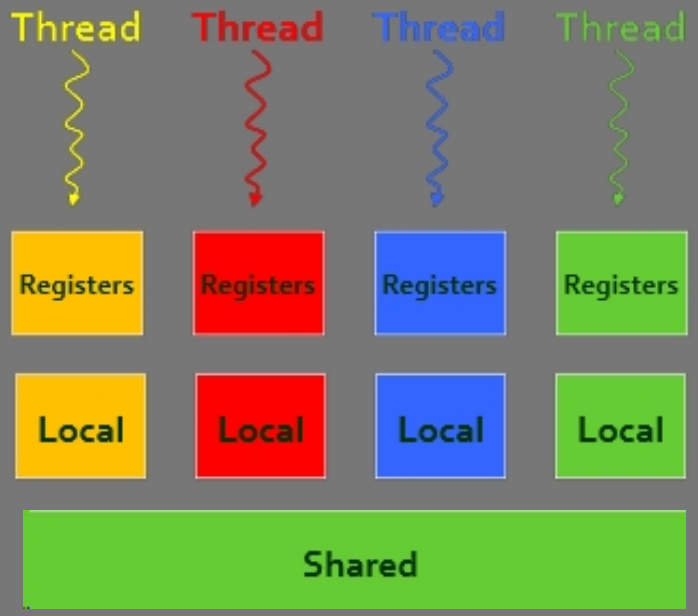
- **Thread**
  - **Registers**
  - **Local memory**



CUDA Model

CUDA Memory

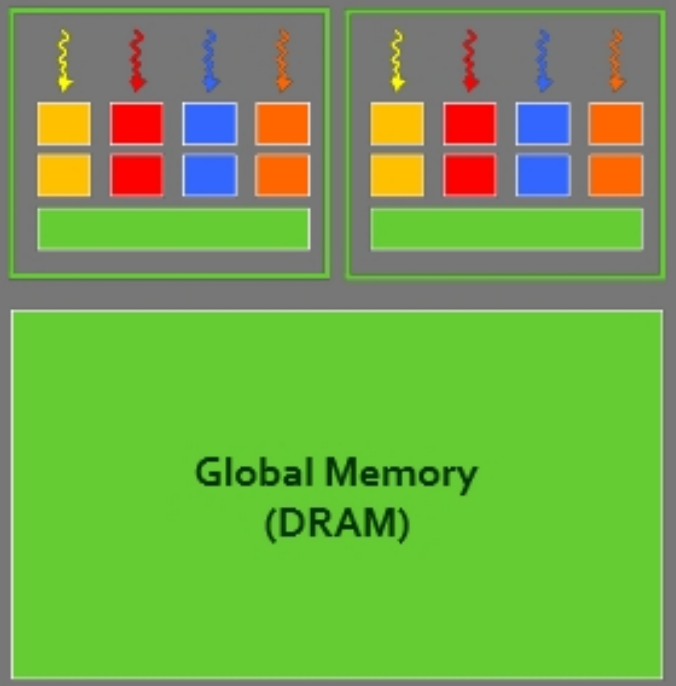
- Thread
  - Registers
  - Local memory
- Thread Block
  - Shared memory



CUDA Model

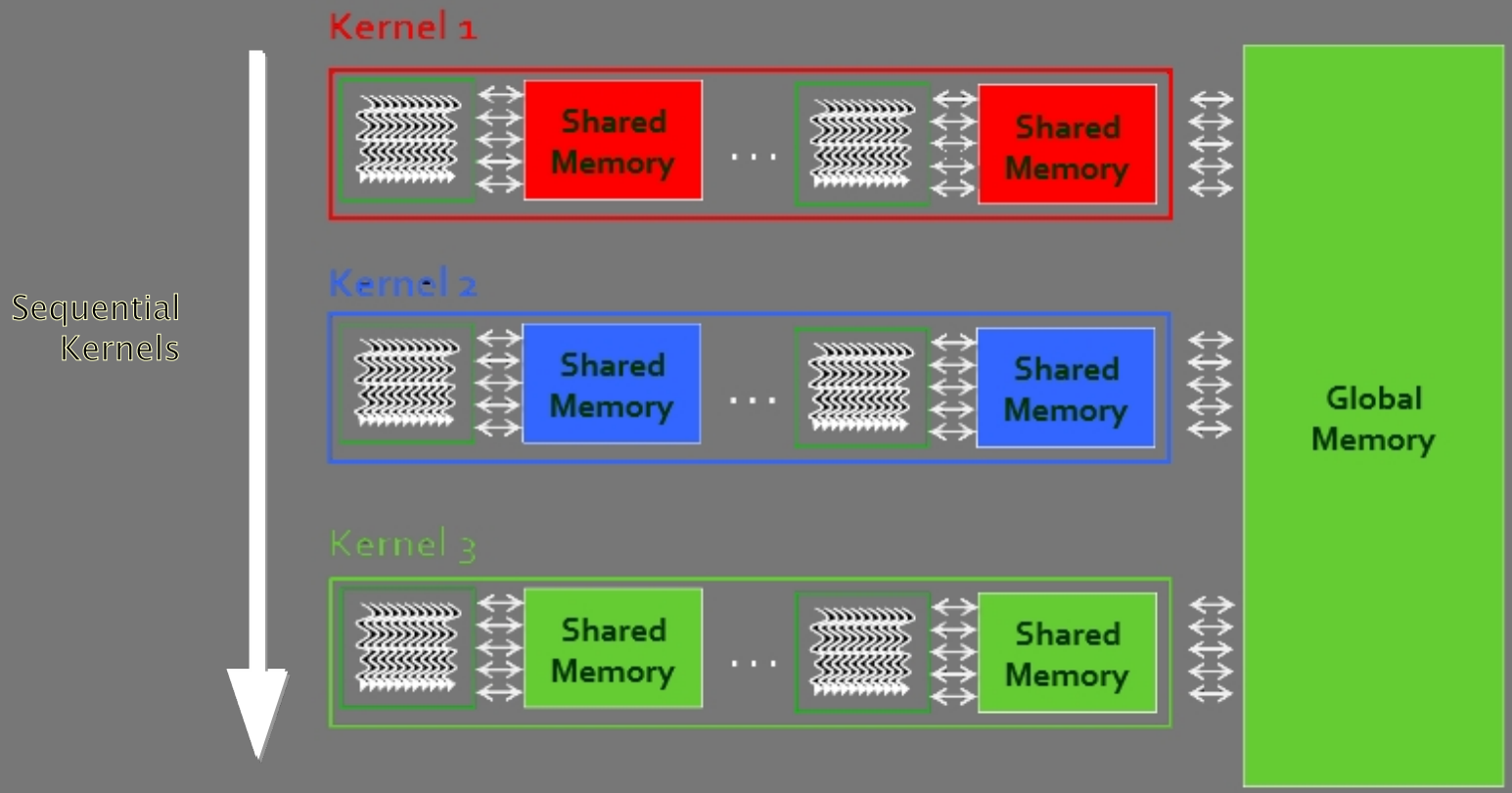
CUDA Memory

- **Thread**
  - Registers
  - Local memory
- **Thread Block**
  - Shared memory
- **All Thread Blocks**
  - Global memory

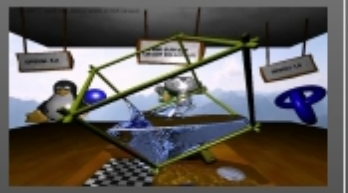


CUDA Model

Communication and Data Persistence



CUDA Applications



Realtime free surface fluid simulation and visuali...

Hybrid Core Acceleration of UWB SIRE Radar Signal ...

Eye-Full Tower: A GPU-based variable multibaseline...

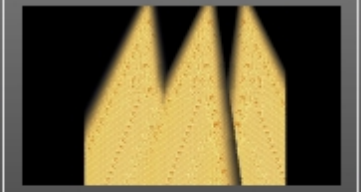
Automatic Generation of Multi-Core Chemical Kernel...

SpofetwraremGPU: Using graphics processing units i...

78 x

nexiwave Speech Indexing

75 x



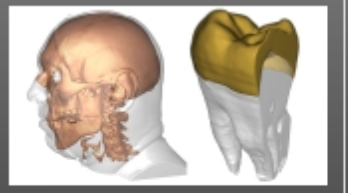
Cellular Automata Evolver

10 x

Performance Characterization of a GPU as a Ubiquit...

High performance GPU radix sorting in CUDA

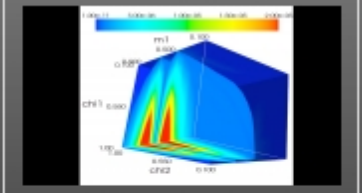
Ultra Fast SOM using CUDA



High-Quality Rendering of Varying Isosurfaces

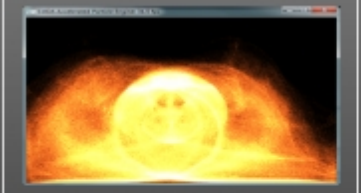
68 x

Parallelising Wavefront Applications on General-Pu...



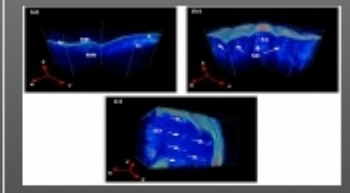
Statistical constraints on binary black hole inspi...

50 x



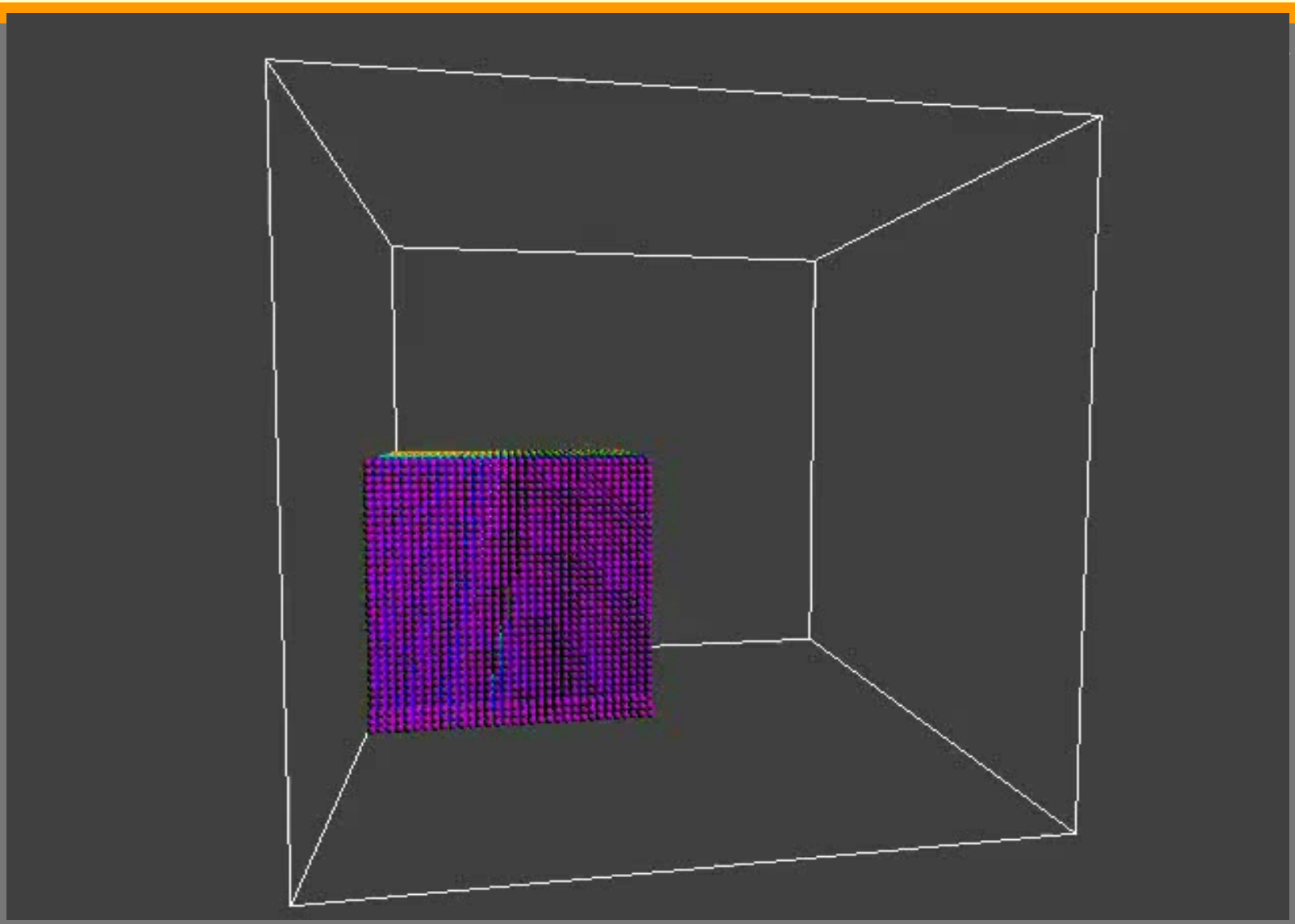
CUDA Accelerated Particle Engine

10 x



Real-time 4D signal processing and visualization u...

CUDA Applications



Summary

- What is GPU
- What is GPU Computing
- Why GPU Computing
- GPU Architecture
- GPU Evolution
- CUDA Model
  - What is CUDA
  - CUDA Kernel
  - Thread Organization
  - Memory

References

Computer Desktop Encyclopedia: graphics pipeline
http://www.answers.com/topic/graphics-pipeline

http://cs.nyu.edu/courses/spring12/CSCI-GA.3033-012/lecture3.pdf

http://en.wikipedia.org/wiki/CUDA

http://www.slideshare.net/ram9a/cuda?from_search=1

http://developer.amd.com/resources/heterogeneous-computing/what-is-heterogeneous-computing/

Thank you

Question?

Feedback?

Suggestion?