MIC-GPU: High-Performance Computing for Medical Imaging on Programmable Graphics Hardware (GPUs)

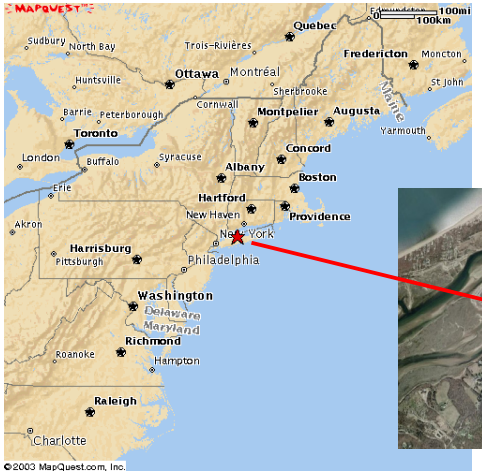
SPIE Medical Imaging

Introduction

Klaus Mueller

Computer Science
Center for Visual Computing
Stony Brook University

Stony Brook University



… on Long Island

about 50 miles from New York City

The Center for Visual Computing

on campus

near Wang Center

in Computer Science Department

Entertainment Graphics: Virtual Realism for the Masses

Computer games need to have:

- realistic appearance of characters and objects
- believable and creative shading, textures, surface details
- realistic physics and kinematics
- effects need to be customizable and interactive

Enabling Technology

PC graphics boards, featuring GPUs:

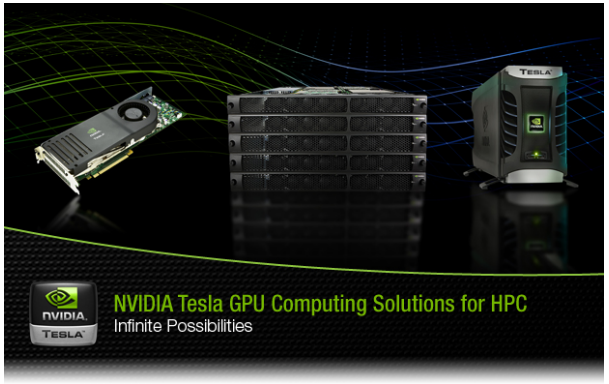
- NVIDIA FX, ATI Radeon
- available at every computer store for less than $500
- set up your PC in less than an hour and play

the latest board (12/07):

NVIDIA GeForce 8800 GTS

Latest Technology Trend

Compute-only (no graphics): NVIDIA Tesla 870



C870 (1 GPU) D870 (2 GPUs) S870 (4 GPUs)

True GPGPU

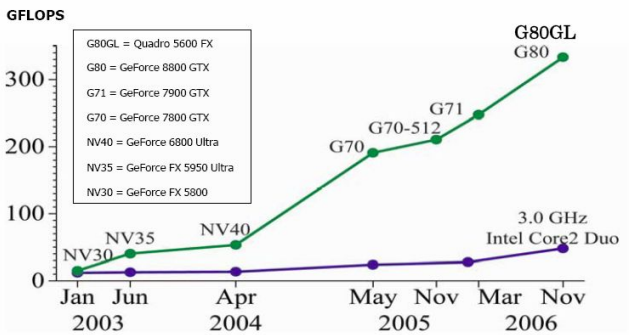
(General Purpose Computing using GPU Technology)

1.5 GB memory per GPU

Incredible Growth

Performance gap GPU / CPU is growing

- currently 1-2 orders of magnitude is achievable (given appropriate programming and problem decomposition)



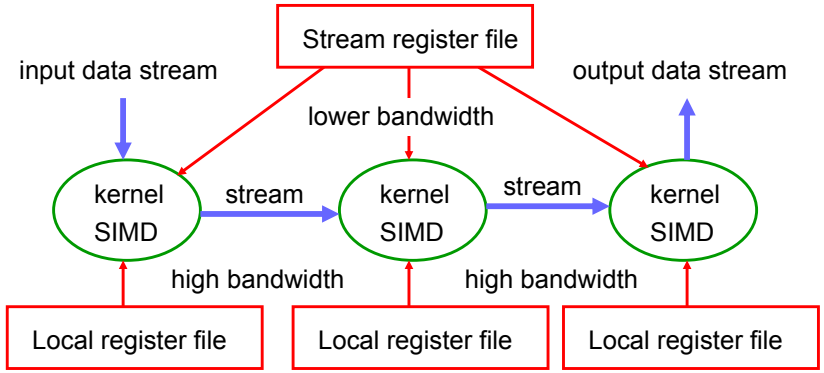
Vital Specs

	GeForce 8800 GTX	GeForce 8800 Ultra
Codename	G80	G80
Release date	11/2006	5/2007
Transistors	681 M (90nm)	681 M (90nm)
Clock speed	1350 MHz	1512 MHz
Processors	128	128
Peak pixel fill rate	36.8 Gigapixels/s	39.1 Gigapixels/s
Pk memory bandwidth	86.4 GB/s (384 bit)	103.7 GB/s (384 bit)
Memory	768 MB	768 MB
Peak performance	520 Gigaflops	576 Gigaflops

Stream Processing

GPUs are *stream processors* [Kapasi '03]

(with some restrictions) [Venkatasubramanian '03]



Stream Processing

Similarity with vector processors (Cray supercomputers, etc.):

- both amortize instruction decode over a long data vector (or stream)
- both expose data parallelism by operating on large data aggregates

1990: 500 M Flops

not drawn to scale

2008: 500 G Flops

Stream Processing

Similarity with vector processors (Cray supercomputers, etc.):

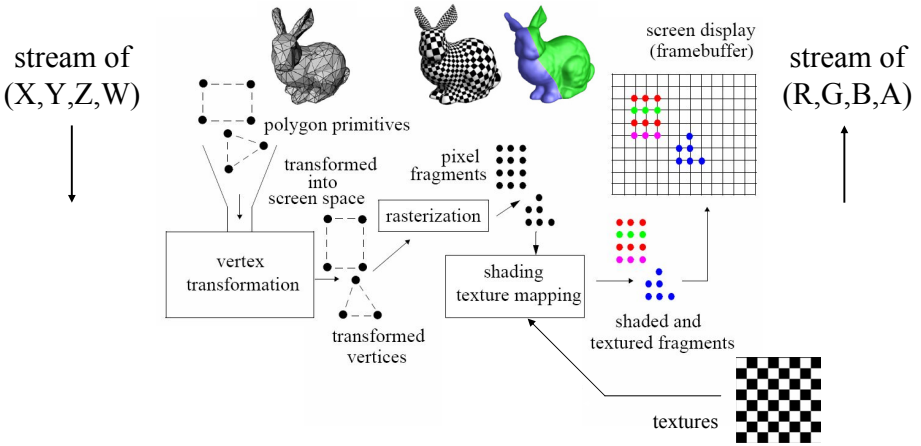
- both amortize instruction decode over a long data vector (or stream)
- both expose data parallelism by operating on large data aggregates

Important generalizations:

- kernels perform all instructions for a single record before moving to the next
  → reduces temporary storage
- stream processors split vector register file into a high-bandwidth (small) local register file and a lower-bandwidth (larger) stream register file
  → makes higher number of stream processors economical

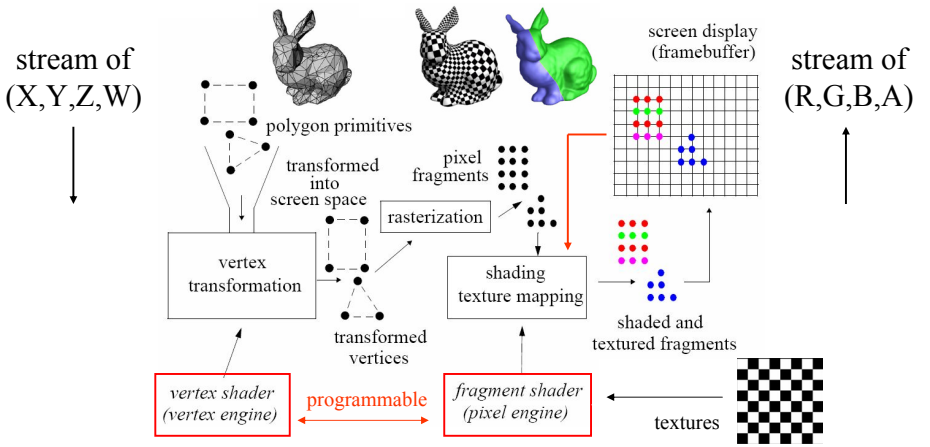
The Graphics Pipeline

Old-style, non-programmable:



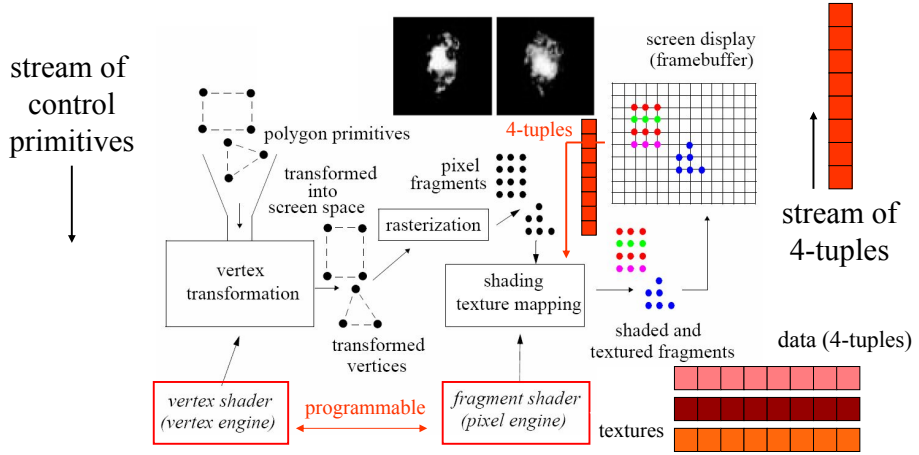
The Graphics Pipeline

Modern, programmable:



The Graphics Pipeline

From a computational view:



GPU vs. CPU

One instruction-decode per kernel stream
- CPU needs a decode for each data item

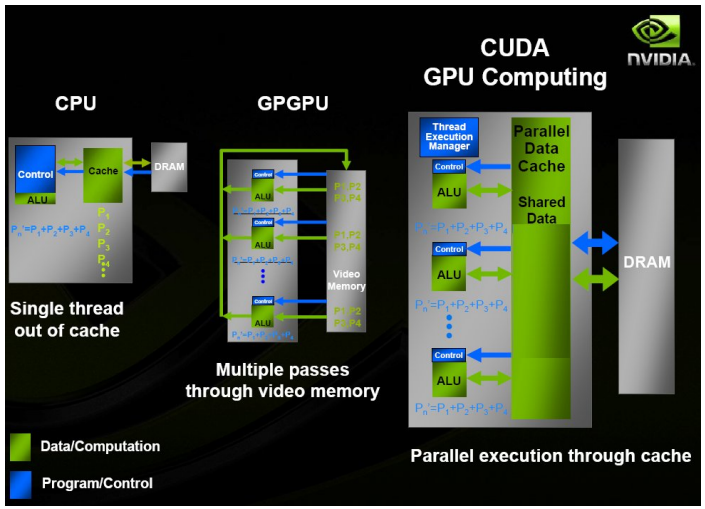
Highly parallel
- GeForce 8800 has 128 x 4-tuple processors (512-way parallel)

Memory very close to processors → fast data transfer
- CPU requires lots of cache logic and communication

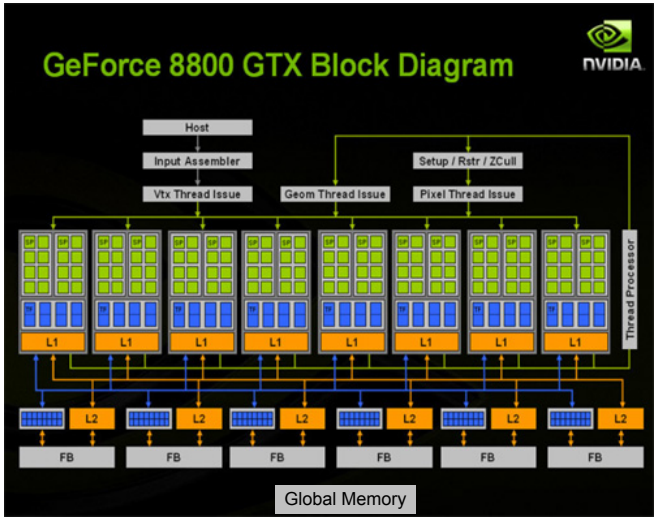
High % of GPU chip real-estate for computing
- small in CPUs (example, 6.5% in Intel Itanium)

In many cases speedups of 1-2 orders of magnitude can be obtained by porting to GPU
- more details on the rules for effective porting later

Comparison: Overview



GPU Architecture: Overview



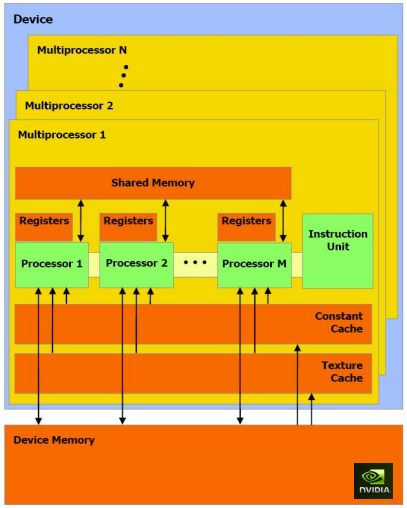
128 processors → 8 multi-processors of 16 processors each

local cache L1 (4k)

shared cache L2 (1M)

DRAM (global memory)

GPU Architecture: Different View



each multiprocessor is a SIMD (Same Instruction, Multiple Data) architecture

equipped with a set of local 32-bit registers (L1 and L2 caches)

the (multi-processor level) shared Constant Cache and Texture Cache are read-only

the (device-level shared) Device Memory (Global Memory) has read-write access and is not cached

GPU Specifics

Four-fold task parallel
- RGBA color vector used for graphics

SIMD (Same Instruction Multiple Data) operation
- multiple parallel pipelines per instruction
- GeForce 8800 has 8 groups of 16 stream processors
- historical note: graphics computations are predominantly SIMD

All standard graphics ops are hardwired
- linear interpolations
- matrix and vector arithmetic (+, -, *)

GPU fully programmable via *fragment programs* or *CUDA*
- at 16 / 32 bit floating point precision
- program length 65k

GPU Specifics

Arithmetic intensity
- the ratio of ALU arithmetic per operand fetched
- needs to be reasonably high, otherwise application is memory-bound

GPU memory 1-2 orders of magnitude slower than GPU processors
- computation often better than table look-ups
- indirections can be expensive

Be aware of GPU 2D-caching protocol (for texture memory)
- data is fetched in 2D tiles (recall graphics bilinear texture filtering)
- promote data locality in 2D tiles

Hide memory latencies by instruction re-ordering
- some support provided by the GPU thread management

GPGPU

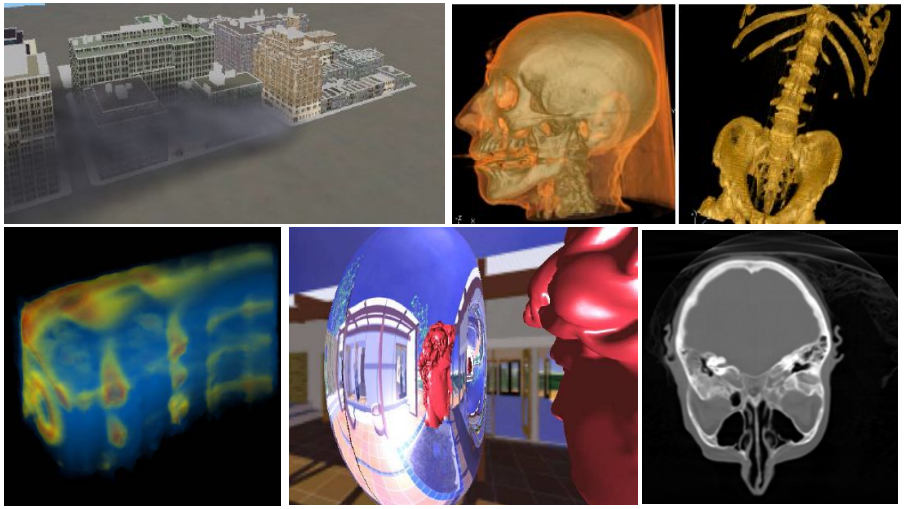
GPGPU = General Purpose Computation on Graphics hardware (GPU)

- massive trend to use GPUs for main stream computing
- see http://www.gpgpu.org

Accelerate

- volume rendering and advanced graphics effects
- computer vision
- scientific computing and simulations
- audio and image & video processing
- database operations, numerical algorithms and sorting
- data compression
- medical imaging
- and many others

GPGPU Example Applications



Latest Hardware Trends

All major hardware vendors (except Intel) are currently seeking to bring SIMD to mainstream computing

AMD/ATI:

- AMD recently bought ATI and just released the Firestream AMD Stream Processor (a repackaged ATI X1900XT GPU board)
- programmable via their new CTM (Close-To-Metal) interface

Nvidia

- released CUDA (Compute Unified Device Architecture) that allows developers to access GPU hardware via C-like language

IBM-Sony-Toshiba

- relased the Cell BE (Broadband Engine) micro processor comprised of 8 tighly-coupled Power processors + one main processor (all together 205 GFlops / 410 for dual processor board))

CUDA

Describes the GPU architecture in much closer detail than what was available before (memory hierarchy, latencies, operation costs, etc)

Promotes computations as SIMD threads, executed in kernels

Exposes details on the GPU memory and thread management

Enables programmers to control all of these in more obvious ways than before with pure graphics primitives (polygon meshes, etc)

But… still requires careful computation flow planning and analysis before coding (when an optimized solution is desired)

Course Schedule

SPIE Medical Imaging

1:30 – 2:00:	Introduction
2:00 – 2:30:	GPU architecture, programming model, and programming facilties (both graphics and CUDA)
2:30 – 3:00:	GPU programming examples (image processing)
	Coffee Break
3:30 – 4:00:	CT reconstruction pipeline components
4:00 – 4:30:	GPU-acceleration of individual components
4:30 – 5:00:	Various CT reconstruction pipelines, load balancing and load estimation
5:00 – 5:30:	Reconstruction visualization and final remarks