Projects

Kayvon Fatahalian

CMU 15-869: Graphics and Imaging Architectures (Fall 2011)

Resources

- There is not a good chip simulator for a modern graphics chip (perhaps that's a project)

- "High-Performance Software Rasterization on GPUs". Laine et al. HPG 2011
  - Full software implementation of the graphics pipeline in CUDA. Fastest available. Great baseline.
  - Source available on Google Code
  - http://research.nvidia.com/publication/high-performance-software-rasterization-gpus
  - See current research ideas (next slide) and evaluate in this context

- NVIDIA Tegra Development Kits
  - Software dev kits may be available for the upcoming Tegra 3 (Kal-El) (we'll have to ask)

- Intel SPMD Program Compiler
  - Generates vector instruction streams from sequential C-like language (motivated by graphics shading languages, but without the graphics-centric concepts)
  - Open source (BSD license)
  - http://ispc.github.com/
  - How fast can a CPU go? Can compiler/runtime techniques effectively hide latency on a CPU?

- Skim through proceedings of:
  - Graphics Hardware (until 2009)

Challenges/themes

- Embracing heterogeneity
  - Developing algorithms designed for heterogeneous systems
  - What simple changes to programmable hardware can be made to accelerate key computations?
- Flipping GPU design inside out (major open problem in graphics systems)
  - One big difference between CPUs and GPUs is what controls what
  - GPU: fixed-function stuff drives programmable stuff (outer loops controlled by hardware)
  - CPU: programmable stuff drives fixed-function stuff
  - GPU approach has worked great, but seems wrong in a hybrid world
- Scheduling
  - Scheduling the graphics pipeline is hard: relies on a lot of heuristics, domain knowledge
    - Could we be more formal? (in the face of dynamic execution?)
    - GRAMPS: A programming model for graphics pipelines [Sugerman TOG 2009][Sanchez ASPLOS 10]
    - Can we quantify the benefit of dropping order preservation?
  - Multi-core, multi-threaded per core, SIMD within a core: dealing with fine-grained parallelism at a scale not present on current CPUs
- Designing good abstractions
  - We've talked about graphics systems as abstract machines (like map-reduce), rather than libraries
  - Does it make sense to explore this strategy in other domains? (what are the triangles, fragments, pixels of X?)
- Understanding workloads

Trending real-time graphics topics

- Issues related to shrinking triangle size
  - Reducing Shading on GPUs Using Quad-Fragment Merging, Fatahalian et al. SIGGRAPH 2010
  - Parallel REYES pipeline implementation
- Stochastic rasterization for accurate camera simulation (rendering with motion and defocus blur)
  - Data-parallel rasterization of micropolygons with motion and defocus blur, Fatahalian et al. HPG 2009
  - Clipless dual-space bounds for faster stochastic rasterization, Laine et. al SIGGRAPH 2011
  - Decoupled sampling for graphics pipelines, Ragan-Kelley et al. Transactions on Graphics 2011
  - Memory system implications when objects start moving around quickly on screen
    - In a rasterizer? In a ray tracer?
- Better anti-aliasing
  - Analytic vs. point-sampling approaches
  - Data-dependent reconstruction [Shirley 2010, 2011][Lehtinen 2011]
  - Programmable pixel operations stage
  - Evaluate quality of screen space vs. object space shading (shade vertices vs. shade fragments)
- Feed-forward (traditional) fragment shading vs. deferred shading
  - Complex bandwidth vs. storage vs. SIMD efficiency tradeoff
  - See Andrew Lauritzen's notes
  - http://bps10.idav.ucdavis.edu/talks/12-lauritzen_DeferredShading_BPS_SIGGRAPH2010.pdf
  - Motion blur/small polygons in a deferred shading system?
- Ray tracing on GPUs or multi-cores (or the combination of the two)
  - Heterogeneous workload (ray tracing + shading)
  - Understanding the Efficiency of Ray Traversal on GPUs. Alia et al. HPG 2009
  - Architecture Considerations for Tracing Incoherent Rays, Alia et al HPG 2010
  - OptiX: a general purpose ray tracing engine, Parker et al. SIGGRAPH 2010