AMD GPU Tools for games development

Holger Gruen
European Developer Relations
AMD Graphics Products Group
holger.gruen@amd.com

Material for many slides was provided by J. Zarge and S. Sowerby from the AMD graphics – Performance Tools - group

Develop conference, 24-26 July 2007
Overview

AMD CPU Tools – a quick reminder
  – CodeAnalyst, APL, ACML, …

AMD GPU Tools
  – GPU PerfStudio, GPU ShaderAnalyzer
  – Tootle
  – AMD content creation tools

GPU Tools demo
  – DX9 case study
  – DX10 demo on DXSDK samples
CPU Tools
CPU Tools: Overview

AMD CodeAnalyst Performance Profiler
- Hotspot detection (various sampling methods)
- Call stack sampling
- Thread profiling, Pipeline simulation, etc. ...
- Plug-in for Microsoft Visual Studio 05

Performance Libraries
- APL (AMD performance library)
- ACML (AMD core math library)
GPU PerfStudio: Overview

Monitors GPU in real-time
- API statistics
- Hardware/driver data of D3D apps

Visualize data in real-time
- in plots
- in bar charts

Client/Server architecture
- Local or remote apps
- Remote app launching

Override rendering states in real-time
GPU PerfStudio Features: Hardware Counters

% Hardware Utilization

% Pixels passed Z-test

Primitive counts

Pixel vs. vertex bound
  - Still makes sense under DX10

etc.
GPU PerfStudio Features: API Statistics

Per-frame API call data
- D3D
- OpenGL

Sorting of API call by
- call count
- call timing
GPU PerfStudio Features: Data Plotting

Marker lines for API state changes
Multiple data series on single plot
Plot properties customizable
GPU PerfStudio Features: Bar Charts

Customizable “alarm” level
Minimum and maximum range values
Data filtering
# GPU PerfStudio Features:
## State Overrides

<table>
<thead>
<tr>
<th>Override</th>
<th>Description/Possible Bottleneck</th>
</tr>
</thead>
<tbody>
<tr>
<td>Force 2x2 Textures</td>
<td>Is texture bandwidth (large textures) affecting performance?</td>
</tr>
<tr>
<td>Force Disable Texture Filtering</td>
<td>Are expensive texture filtering modes affecting performance?</td>
</tr>
<tr>
<td>Force 1x1 Scissor Region</td>
<td>Identifies vertex processing bottlenecks (by removing most pixel processing)</td>
</tr>
<tr>
<td>Force Simple Pixel Shaders</td>
<td>Identifies expensive pixel shaders</td>
</tr>
<tr>
<td>Force Skip Draw*Prim Calls</td>
<td>Identifies non-GPU bottlenecks (by removing most 3D graphics work)</td>
</tr>
<tr>
<td>Force Z Test Enable</td>
<td>Identifies z-order performance issues</td>
</tr>
<tr>
<td>Force Z Write Enable</td>
<td>Identifies z-order performance issues</td>
</tr>
<tr>
<td>Force Alpha Blend Enable</td>
<td>Identifies alpha-blending performance issues</td>
</tr>
<tr>
<td>Force Alpha Test Enable</td>
<td>Can identify problems related to early Z test</td>
</tr>
<tr>
<td>Force Cull Mode</td>
<td>Can show culling efficiency</td>
</tr>
<tr>
<td>Force Fill Mode</td>
<td>Used for debugging and identifying vertex density</td>
</tr>
</tbody>
</table>
GPU PerfStudio: Upcoming Features

DX10/Vista/HD2000 support
- ability to analyze cutting edge applications on the newest hardware

Render target and non-RT state overrides
- Especially useful for image-space effects (depth of field, glows, render to texture)

Data filters – average median, min, max , derivative
- For gaining additional insight into your data

Plot lines for user defined markers
- Demonstrate the effect of state override on performance data
Flexible bar charts
  - Control colors, sizes, alarm, range

Flexible table cell formatting
  - Control cell size, color, font

Application remembers settings
  - recent sessions, recent server machines, recent apps, etc.

Selectable anti-aliasing
  - Fine tune the look and performance of GPU PerfStudio
GPU ShaderAnalyzer
GPU ShaderAnalyzer: Feature Overview

Shader performance analysis tool
Shader tuning environment
Instant perf. feedback as you tune your shaders
**GSA Features: Performance Analysis**

Predicts shader perf. on range of AMD GPUs.

Analyzes compiled hardware instruction stream

- Analysis is tied to specific Catalyst driver releases.

Displays estimated cycle count & ALU:Texture ratio

Color codes ALU:Tex ratio for shader for

- ALU bound, Texture bound, Interpolator bound
GSA Features: Performance Analysis (cont.)

Considers static & dynamic flow control.

Considers the cost of each side of a branch

Calculate the minimum, maximum & average cycles

Factors in the expected flow control coherence

Currently only average cycles are displayed in GUI
  - but min & max from command line and selectable in new versions

Estimates cost of texturing for bi-, tri-linear & aniso
  - Cost based on typical texture fetch cost, not theoretical maximum
GSA Features: Hardware Disassembly

View the actual shader as executed by the hardware
Shows the hardware shader optimized by the SC
See where your shader performance is going
Can also display D3D shader disassembly
GSA Features: Supported Shader Formats

DX9 Pixel\Vertex Shaders
SM 1.1 – SM 3.0 Assembly Shaders
SM 2.0 – SM 3.0 HLSL Shaders
DX10 Pixel\Vertex\Geometry Shaders
GLSL Pixel\Vertex Shaders
arb_fp\arb_vp programs
GSA Features:
Other features

Options dialog to configure
- HLSL compiler to use
- ATI Shader Compiler Version
- GPUs to analyze performance for
- Options that control code analysis

Command line support
- performance analysis & hardware disassembly also available from command line – as mentioned earlier
- Analyze a single shader or a directory tree
- Output analysis to .csv file for further analysis within MS Excel
Tootle: Overview

A Triangle Order Optimization Tool
- Improves vertex-cache hit rate
- Reduces overdraw
- View independent

Library to integrate into your tool-chain

Simple to use and free
Tootle: Background

Based on I3D 06 paper by Nehab\Barczak\Sander

Uses D3DXOptimizeMesh for vtx cache optim.

Uses D3D for overdraw measurement

Example Scene: 70k polygons, 10 materials

- Reduced overdraw by factor of two.
- 3-7% performance increase compared to D3DXOptimizeMesh
Tootle: Overdraw Reduction
AMD Content Creation Tools
AMD Content Creation Tools

RenderMonkey
- Shader development environment
- Supports HLSL, D3D asm and GLSL

The Compressonator
- Tool for compressing textures
- Creates mip-map levels
- DX10 supported

CubeMapGen
- Creates filtered seamless cube maps
- Uses angular extent filtering

NormalMapper
- Automatic normal map generation tool
GPU Tools demo: DX9 case study
Case Study: Settlers 6 (working title)

Developer is BlueByte
- Development started spring ´05
- Will be released this summer

Empire building RTS game
- Set in a medieval world
- Has a focus on city building
- Warfare part of the game

Single & Multi player modes
- Various multi player modes
- Story based and free-play maps

Complex graphics engine
- Highly dynamic world
- Lots of foliage
Why were the tools used?

The Tools are not invasive!

- No changes to your source code
- No additional risk even late during a project
- No additional Q&A 😊
- The tools are free!!

Tools would allow most efficient way to

- Assess impact of rendering techniques for new scene elements
- Only one scene element covered by this study
GPU PerfStudio: Rendering flowers - no instancing
GPU PerfStudio:
Rendering flowers - instancing
GPU PerfStudio: Rendering flowers – vertex limited

Hardware Counters

- % Hardware Utilization: 90.912636
- % Vertex Wait for Execution: 5.021636
- % Pixel Wait for Execution: 76.222565
- Pre-clip Primitives: 108564
- Post-clip Primitives: 38835
- % Blended Pixels: 42.075352
- % Pixels Passed: 100.000000
- Overdraw: 0.583057
An experiment

Lot’s of pixels are waiting for vertices
- Reasons are complex scattering computations in vertex program
- Could pixels be more expensive at certain low resolutions?

GSA tells us PS could abide for more ALU on x1900
- Especially for Trilinear and Aniso
- There is a chance we can improve speed

Let’s move scattering computations from VS to PS
- This is just an experiment!
- Won’t go into details about the shader changes!
Reassessing the situation

Ratios went up for X1900

- It is what we would expect 😊
- We are now ALU limited
- Good for future GPUs 😊

We are now green even for Aniso

Let’s check what changed

- Let’s run GPU PerfStudio again ...
Let’s run GPU PerfStudio again …

The experiment was a success …
GPU Tools demo: DX10 SDK samples
GPU PerfStudio on Vista/DX10
Questions?

Holger Gruen
European Developer Relations
AMD Graphics Products Group
holger.gruen@amd.com