OpenGL Performance Tools

Sébastien Dominé
NVIDIA Developer Tools
Agenda

- NVShaderPerf 71.84
- NVPerfKIT 1.0
- Conclusion & Q&A
v2f BumpReflectVS(a2v IN, uniform float4x4 WorldViewProj, uniform float4x4 World, uniform float4x4 ViewIT)

{ v2f OUT;
// Position in screen space.
OUT.Position = mul(IN.Position, WorldViewProj);
// pass texture coordinates for fetching the normal map
OUT.TexCoord.xyz = IN.TexCoord;
OUT.TexCoord.w = 1.0;
// compute the 4x4 transform from tangent space to object space
float3x3 TangentToObjSpace;
TangentToObjSpace[0] = float3(IN.Tangent.x, IN.Binormal.x, IN.Normal.x);
TangentToObjSpace[1] = float3(IN.Tangent.y, IN.Binormal.y, IN.Normal.y);
TangentToObjSpace[2] = float3(IN.Tangent.z, IN.Binormal.z, IN.Normal.z);
OUT.TexCoord1.x = dot(World[0].xyz, TangentToObjSpace[0]);
OUT.TexCoord1.y = dot(World[1].xyz, TangentToObjSpace[0]);
OUT.TexCoord1.z = dot(World[2].xyz, TangentToObjSpace[0]);
OUT.TexCoord2.x = dot(World[0].xyz, TangentToObjSpace[1]);
OUT.TexCoord2.y = dot(World[1].xyz, TangentToObjSpace[1]);
OUT.TexCoord2.z = dot(World[2].xyz, TangentToObjSpace[1]);
OUT.TexCoord3.x = dot(World[0].xyz, TangentToObjSpace[2]);
OUT.TexCoord3.y = dot(World[1].xyz, TangentToObjSpace[2]);
OUT.TexCoord3.z = dot(World[2].xyz, TangentToObjSpace[2]);

// compute the eye vector (going from shaded point to eye) in cube space
float4 eyeVector = worldPos -- ViewIT[3];

OUT.TexCoord1.w = eyeVector.x;
OUT.TexCoord2.w = eyeVector.y;
OUT.TexCoord3.w = eyeVector.z;
return OUT;
}

 NVShaderPerf 71.84

Outputs:
• Assembly code
• # of cycles
• # of temporary registers
• Pixel throughput
• Forces all fp16 and all fp32

Inputs:
• GLSL (fragments)
• FF1.0
• ARBfp1.0
• Cg

GPU Arch:
• GeForce FX
• GeForce 6 Series
• Quadro FX

Glitchy Technologies

NVIDIA

GDC 2005
NVShaderPerf - Next...

• Adding more complete support for vertex programs
  – Vertex Throughput
  – GLSL Vertex Scheduling

• Support for multiple driver versions from one release
NVPerfKit 1.0

- Complete Performance Instrumentation Solution
  - Instrumented Driver
  - NVIDIA Developer Control Panel
  - Supports PDH (Performance Data Helper)
  - Code Samples for OpenGL and Direct3D
  - Requires Instrumented Applications to be Authorized
**NVPerfKit**

- **Instrumented Driver**
  - Exposes GPU and Driver Performance Counters
  - Supports OpenGL
  - Supports SLI Counters
  - Requires GeForce FX and 6 Series

- **NVIDIA Developer Control Panel**
  - Required to enable the counters to be sampled
  - Easy GUI to preset counter bundles
NVPerfKit

- OpenGL Driver Counters:
  - FPS & frame time (1/FPS)
  - AGP texture, VBO, and total memory used
  - Video texture, VBO, and total memory used
  - driver sleep time (driver waits for GPU),
NVPerfKit

- HW/GPU counters:
  - gpu idle
  - pixel shader utilization
  - vertex attribute count
  - vertex shader utilization
  - texture waits for shader
  - shader waits for texture
  - shader waits for rop
  - FastZ utilization (UltraShadow)
  - pixel, vertex, triangle, primitive & culled primitive counts
NVPerfKit

- **Microsoft Performance Data Helper (PDH)**
  - WMI (Windows Management and Instrumentation)
  - VTune, Perfmon, ...

- Access counters in your own application

```c
// Setup
PDH_HQUERY hQuery;
PDH_COUNTER hCounter;

PDH_STATUS status = PdhOpenQuery(0,0,&hQuery);
PdhAddCounter(hQuery, "\\NVIDIA GPU Performance(GPU0/% gpu_idle)\\GPU Counter Value",0,&hCounter));

// Periodically...
PDH_STATUS status = PdhCollectQueryData(hQuery);
PDH_FMT_COUNTERVALUE cvValue;
PdhGetFormattedCounterValue(hCounter,PDH_FMT_DOUBLE|PDH_FMT_NOCAP100|PDH_FMT_NOSCALE,0,&cvValue)
```

GDC 2005
NvPerfKit

Performance

- Console Root
  - System Monitor
  - Performance Logs and Alerts

Graph showing performance metrics such as GPU Counter Value, % gpu_idle, etc.

- Last: 34.968
- Average: 62.919
- Minimum: 23.944
- Maximum: 99.999

Duration: 1:40

<table>
<thead>
<tr>
<th>Color</th>
<th>Scale</th>
<th>Counter</th>
<th>Instance</th>
<th>Parent</th>
<th>Object</th>
<th>Computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.000</td>
<td>GPU</td>
<td>Counter Value</td>
<td>% gpu_idle</td>
<td>GPU</td>
<td>NVIDIA GPU Performance</td>
<td>\SDOMINE-P...</td>
</tr>
<tr>
<td>1.000</td>
<td>GPU</td>
<td>Counter Value</td>
<td>% primitive_count</td>
<td>GPU0</td>
<td>NVIDIA GPU Performance</td>
<td>\SDOMINE-P...</td>
</tr>
<tr>
<td>1.000</td>
<td>OGL</td>
<td>Counter Value</td>
<td>% OGL % driver waiting</td>
<td>CPU</td>
<td>NVIDIA OpenGL Driver</td>
<td>\SDOMINE-P...</td>
</tr>
<tr>
<td>1.000</td>
<td>OGL</td>
<td>Counter Value</td>
<td>% OGL % pushbuf used</td>
<td>CPU</td>
<td>NVIDIA OpenGL Driver</td>
<td>\SDOMINE-P...</td>
</tr>
<tr>
<td>1.000</td>
<td>OGL</td>
<td>Counter Value</td>
<td>OGL FPS</td>
<td>CPU</td>
<td>NVIDIA OpenGL Driver</td>
<td>\SDOMINE-P...</td>
</tr>
<tr>
<td>1.000</td>
<td>OGL</td>
<td>Counter Value</td>
<td>OGL frame mSec sleeping</td>
<td>CPU</td>
<td>NVIDIA OpenGL Driver</td>
<td>\SDOMINE-P...</td>
</tr>
<tr>
<td>1.000</td>
<td>OGL</td>
<td>Counter Value</td>
<td>OGL frame time mSec</td>
<td>CPU</td>
<td>NVIDIA OpenGL Driver</td>
<td>\SDOMINE-P...</td>
</tr>
</tbody>
</table>
Why a Security Mechanism?
- No Reverse Engineering
- Have to own the application’s code

How does that work?
- Application are authenticated
- Lightweight runtime checks
  - Heartbeat regularly checks for validity
NVPerfKit

• Schedule
  – Registered Developer Early Beta Access: 3/16/2005
  – Registered Developer Beta: 3/23/2005
  – Registered Developer Release Target: 4/7/2005
Conclusion

- **Shader Performance Regression Tools**
- **Complete Instrumentation Solution**
  - NvPerfHUD for OGL
    - Just do the HUD rendering in your application
    - use PDH for the rest
- **FX Composer 2.0 will support OpenGL**
  - GLSL, Cg, CgFX and NVshaderPerf
- **...for everything else, there is Graphics Remedy’s gDEBugger**
The Source for GPU Programming

developer.nvidia.com

- Latest News
- Developer Events Calendar
- Technical Documentation
- Conference Presentations
- GPU Programming Guide
- Powerful Tools, SDKs and more ...

Join our FREE registered developer program for early access to NVIDIA drivers, cutting edge tools, online support forums, and more.

developer.nvidia.com

©2004 NVIDIA Corporation. NVIDIA, and the NVIDIA logo are trademarks and/or registered trademarks of NVIDIA Corporation. Nalu is ©2004 NVIDIA Corporation. All rights reserved.