Beyond Performance: Introducing NVIDIA’s New Graphics Debugger

Aurelio Reis, Director of Engineering, Graphics Developer Tools
Jeff Kiel, Engineering Manager, Graphics Developer Tools

March 22

Booth #223 - South Hall
www.nvidia.com/GDC
Agenda

- Overview
- Demo
- Key Takeaways
- Q&A
Overview
The Story So Far…

- Nsight: Visual Studio Edition
  - Fully integrated into IDE
  - Graphics
  - Compute (CUDA)
  - 9+ years of learnings

- Ideal workflow for developers, but...
  - Reliant on VS ecosystem
  - Dependencies on compute for release

How can we improve?
Introducing: Nsight Graphics
Key Pillars

**Debugging**
- C++ Serialized Captures
- Event Timeline (Scrubber)
- Shader Editing
- Pixel History
- Resources Viewer
- Geometry Viewer
- API Inspection

**Profiling**
- Range Profiler
- HW Perf Counters
- Shader Stats
- Event Timings
- Counter Selector
- API Stats
- GPU Trace

**Ray Tracing (RTX)**
- DXR Capture/Replay
- DXR Frame Debugger
- Shader Bind Table View
Key Pillars

Debugging

Profiling

Ray Tracing (RTX)
Key Pillars

Debugging

Profiling

Ray Tracing (RTX)
Key Pillars

Debugging
- C++ Serialized Captures
- Event Timeline (Scrubber)
- Shader Editing
- Pixel History
- Resources Viewer
- Geometry Viewer
- API Inspection

Profiling
- DXR Capture/Replay
- DXR Frame Debugger
- Shader Bind Table View

Ray Tracing (RTX)
- Ray Tracing
- Event Timings
- Counter Selector
- API Stats
- GPU Trace
- HW Perf Counters
- Shader Stats

Hardware
- SDK
- DirectX
- Vulkan
- OpenGL
- Metal
- WAPI
Developer Choice

Nsight: VSE

+ Visual Studio integration
+ Compute tools
+ Tracing tools

Nsight Graphics

+ More frequent releases
+ Activities system
+ Advanced profiling
Demo

1. Nsight Graphics Tour
2. DXR Debugging
3. Optimizing Async Compute
Demo 1 - Nsight Graphics Tour
Nsight Graphics Tour

New, stand alone shell!

Connection settings

Project system & artifact management
Nsight Graphics Tour

Select Platform…

…add application parameters…

…select Activity…

…Launch!
Nsight Graphics Tour
Targeted UI Layout Per Activity

Deep dive into frame rendering and API analysis
Automated throughput analysis
Create C++ Capture for standalone exe reps
Analysis of shader unit utilization…coming soon!
Nsight Graphics Tour

- Real time performance metrics
- Scrub through the scene, better understand GPU rendering
- Live capture: return to your application at will
Nsight Graphics Tour

Event List: API event trace

Scrubber: break down frame construction

Current Target View: visualize render targets

API Inspector: full API state inspection
Nsight Graphics Tour

Events View: filter to find problematic API usage

Frequent shader changes can impact performance
Pipepline Navigator: state categorized by GPU location

State laid out for easy inspection

Hyper-linked UI for easy navigation

Nsight Graphics Tour
Nsight Graphics Tour

Inspect all resource types, views, etc.

Scrub through buffer revisions

Initiate Pixel History!
Nsight Graphics Tour

Show all fragments that potentially touched a pixel location

Inspect RT Before, Fragment, and RT After color values

Understand why fragments DON’T land in the frame buffer, like back facing or Z-culled
DXR Debugging
DXR Debugging

- Inspect bound Ray/Miss/Hit Shaders
- Visualize Root Parameters/Arguments and associated data values
Demo 3 - Profiling
“The Peak-Performance Analysis Method for Optimizing Any GPU Workload”
GPU Trace: How filled are my SMs?

- Visualize warp occupancy by workload type
- Display performance markers, command lists, individual draw calls, etc.
- Statistics for current selection
- Determine when Graphics/Compute pipe is idle vs just shader units
GPU Trace: How filled are my SMs?

Note graphics and compute work occurring on a single, Direct Queue

Not achieving optimal occupancy of ~65-70%

Frame time is just over 8ms

Solution? Use a Compute Queue
Construct Compute Queue

```c
#include <d3d12.h>

D3D12_COMMAND_QUEUE_DESC computeQueueDesc =
{
    D3D12_COMMAND_LIST_TYPE_COMPUTE, // Type
    0, // Priority
    D3D12_COMMAND_QUEUE_FLAG_NONE, // Flags
    0x0 // NodeMask
};

// Create a compute command queue
hResult = pD3DDevice->CreateCommandQueue(&computeQueueDesc, __uuidof(ID3D12CommandQueue), (void**)&pD3DComputeCommandQueue);

// Create a compute command allocator
hResult = pD3DDevice->CreateCommandAllocator(D3D12_COMMAND_LIST_TYPE_COMPUTE, __uuidof(ID3D12CommandAllocator),
     (void**)&pComputeCommandAllocator);

// Create a compute command list
hResult = pD3DDevice->CreateCommandList(0x1, D3D12_COMMAND_LIST_TYPE_COMPUTE, pComputeCommandAllocator, NULL,
    __uuidof(ID3D12GraphicsCommandList), (void**)&pComputeCommandList);
```
Direct Queue ensures work ordering.

Additional Compute Queue allows for Async Compute to overlap Graphics.
GPU Trace: How filled are my SMs?

Compute Queue visualizes from the top

Queues are serialized...fences!
No (Real) Async: Debug with Nsight Graphics
GPU Trace: Much Better!

Graphics & Compute now overlap…better shader unit utilization!

Frame time now just over 6ms...~2ms saved!
But wait... why is compute not full?

Compute work doesn't "fill" the GPU... why?

Comes from "Primary Rays" Performance Marker Region
GPU Not Full: Debug with Nsight Graphics...

- Find "Primary Rays" section in Scrubber
- Identify shader in Scrubber/Source View
GPU Not Full: Debug with Nsight Graphics...

Shader uses 32 registers

32 registers limits occupancy to 48 of 64 possible warps...~80%
GPU Throughput: SOL

SOL = Speed of Light

< ~10%, unit just not taxed...

Unit not achieving maximum throughput, analyze other metrics to determine why
GPU Throughput: SOL

SOL = Speed of Light

Grey area: can try to make unit more efficient, or could try and reduce work
GPU Throughput: SOL
SOL = Speed of Light

Unit is efficient but near max throughput, will likely need to shift work to other units improve performance
Understanding GPU Performance

Range Profiler

- GPU time for every draw/dispatch
- Elapsed time for performance marker ranges
- Pipeline throughput/SOL values
Understanding GPU Performance

Range Profiler

Overall frame is mainly texture limited, but better to dig into ranges that have similar workloads

TEX: 26.9% | SM: 18.2% | L2: 12.6% | DRAM: 8.6% | CROP: 7.4%

<table>
<thead>
<tr>
<th>Top SQLs</th>
<th>TEX: 26.4%</th>
<th>SM: 13.3%</th>
<th>L2: 12.6%</th>
<th>DRAM: 8.6%</th>
<th>CROP: 7.4%</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPU idle</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSE Stall Cycles</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SM Active</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SM Active Min/Max Delta</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SM Issue Utilization Per Active Cycle</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SM Occupancy (Active Warps Per Active Cycle)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
Understanding GPU Performance

Range Profiler

Select largest/most expensive marker range in scene…

This range is mainly texture and shader unit limited…but dig a little deeper…

TEX: 45.2% | SM: 29.5% | L2: 23.1% | DRAM: 8.4% | CROP: 6.0%
Understanding GPU Performance

G-Buffer resolve…

SM Active Threads Per Instruction Executed = 73.5%
SM Warp Stall Long Scoreboard = 51.6%
Shader unit waiting for texture fetches to return

Texture & Shader throughput low…

SM Active Min/Max Delta @ 24.1% => shader divergence
Understanding GPU Performance

Shadow Map Rendering, typically vertex and z limited

Top SOLs – typical signature for vertex limited work

PD: Primitive Distributor (vertex work)
VPC: Viewpore Cull/Clip (vertex work)
ZROP: Z “blending” (Z buffering)

PD: 61.8% | VPC: 28.9% | ZROP: 14.5% | L2: 7.8% | DRAM: 6.1%
Understanding GPU Performance

Render Gbuffer typically color blending texture, and shader limited

Top SOLs
CROP/ZROP: Blending unit (blending the 6 “fat” render targets & depth)
TEX: Texture unit (reading in diffuse textures, etc.)
SM: Shader unit (calculating Gbuffer values)

CROP: 43.4% | ZROP: 27.7% | TEX: 24.0% | SM: 19.0% | DRAM: 15.1%
Key Takeaways
Summary

▪ **Nsight Graphics** is *the* next-gen graphics debugger
  - DXR Debugging
  - Range Profiler
  - GPU Trace

▪ **NVIDIA DevTools** is committed to…
  - increasing release frequency
  - solving developer pain points
  - improving developer productivity
Microsoft PIX for Windows

- Close collaboration to help make the best tools for our developers
  - Work history buffer i.e. timing activity
  - HW Performance Counters
  - GPU Occupancy Viewer
  - Performance warnings
  - DXR visualization
The Future of Nsight Graphics

- **H1 ‘18**
  - Vulkan 1.1 support
  - Linux Support
  - Improvements to Range Profiler
  - GPU Trace 1.0
  - DXR improvements

- **H2 ‘18**
  - GPU Trace for Vulkan
  - Pixel History (DX12 & Vulkan)
  - DXR Profiling
What’s Next?

- Visit our Booth (#223) to check out *Nsight Graphics, DXR Debugging* and *GPU Trace*
- Check out Louis Bavoil’s talk in 30 minutes to learn more about performance triaging with the *Range Profiler*
- Watch our Videos: [http://j.mp/nvidia-devtools-videos](http://j.mp/nvidia-devtools-videos)

Join our Early Access Program and get
*Nsight Graphics* here:

[https://j.mp/ngfx](https://j.mp/ngfx)

We’re Hiring!


“NVIDIA has made a major jump from using technology to create entertainment to using it to alter our reality and change the world.”

- Rob Enderle, Industry Analyst
Thank you!

Questions?

Booth #223 - South Hall
www.nvidia.com/GDC

Need to contact us?
NsightGraphics@nvidia.com

Get Nsight Graphics here: https://j.mp/ngfx
(https://developer.nvidia.com/nsight-graphics)
## NVIDIA Sessions

<table>
<thead>
<tr>
<th>Day</th>
<th>Time</th>
<th>Room</th>
<th>Speaker</th>
<th>Title</th>
</tr>
</thead>
</table>
| Thurs Mar 21 | 17:30-18:30   | 3022    | Louis Bavoil  
Principal Engineer                         | Fixing the Hyperdrive - Maximizing Rendering Performance on NVIDIA GPUs |
| Fri Mar 23  | 10:00-11:00   | 3022    | Nuno Subtil  
Senior Developer Technology Engineer         | NVIDIA Vulkan Update                                                  |
| Fri Mar 23  | 11:30-12:00   | 3001, 3003 | Alex Dunn   
Senior Developer Technology Engineer          | Aftermath – Advances in GPU Crash Debugging                           |
| Fri Mar 23  | 12:15-13:15   | 3022    | Bryan Dudash  
Senior Manager Developer Technology            | Capture Amazing Content with NVIDIA Ansel Photo Mode and Highlights Video Capture Tool |
| Fri Mar 23  | 13:30-14:30   | 3022    | Evan Hart, Principal Engineer                | Advances in the HDR Eco-System                                       |
| Fri Mar 23  | 15:00-16:00   | 3022    | Cem Cebenoyan  
Director of Engineering                       | Accelerating your VR Games with VRWorks                               |

Need to contact us?  
NsightGraphics@nvidia.com

Get Nsight Graphics here: [https://j.mp/ngfx](https://j.mp/ngfx)  
(https://developer.nvidia.com/nsight-graphics)