
intel
GAME DEVELOPERS
CONFERENCE
IT DOESN'T HAVE TO BE HARD:
HOW TO FIX YOUR
PERFORMANCE WOES
Carlos A Dominguez Caballero (Intel®)

Agenda

- Introduction
- Game profiling workflow overview
- Step through profiling workflow
- Common bottleneck identification
- Case Study: Optimizing Unity's 3D Game Kit

Scaling Graphics Performance Effectively

My super awesome game

Scaling Graphics Performance Effectively

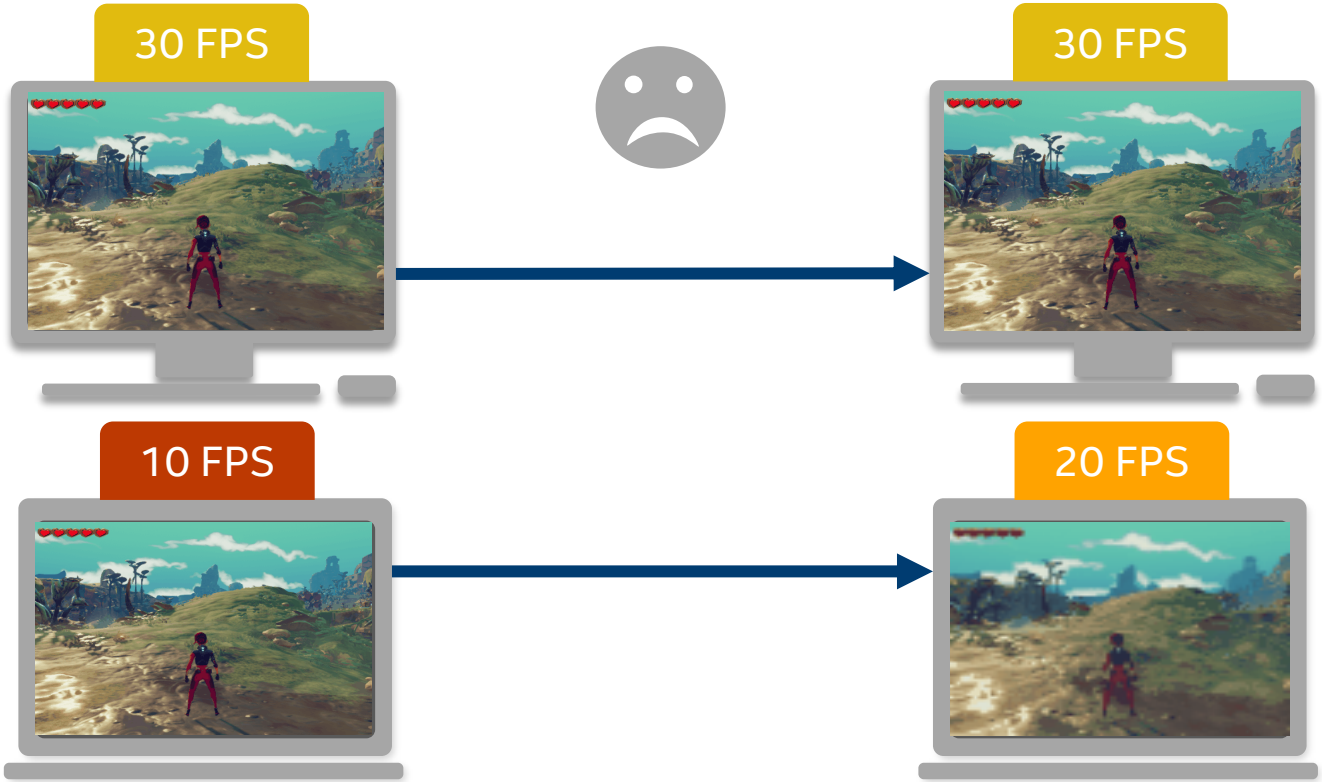
My super awesome game

Scaling Graphics Performance Effectively

My super awesome game

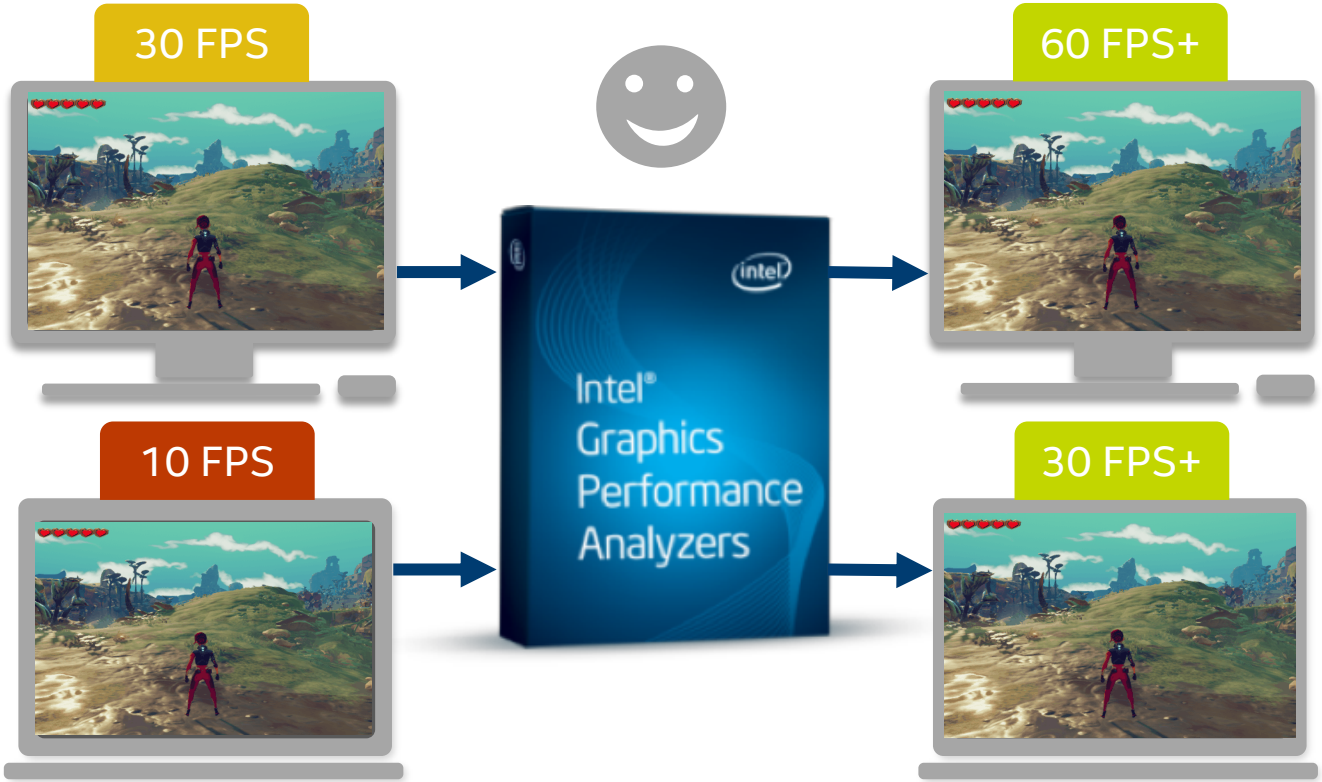
Scaling Graphics Performance Effectively

My super awesome game

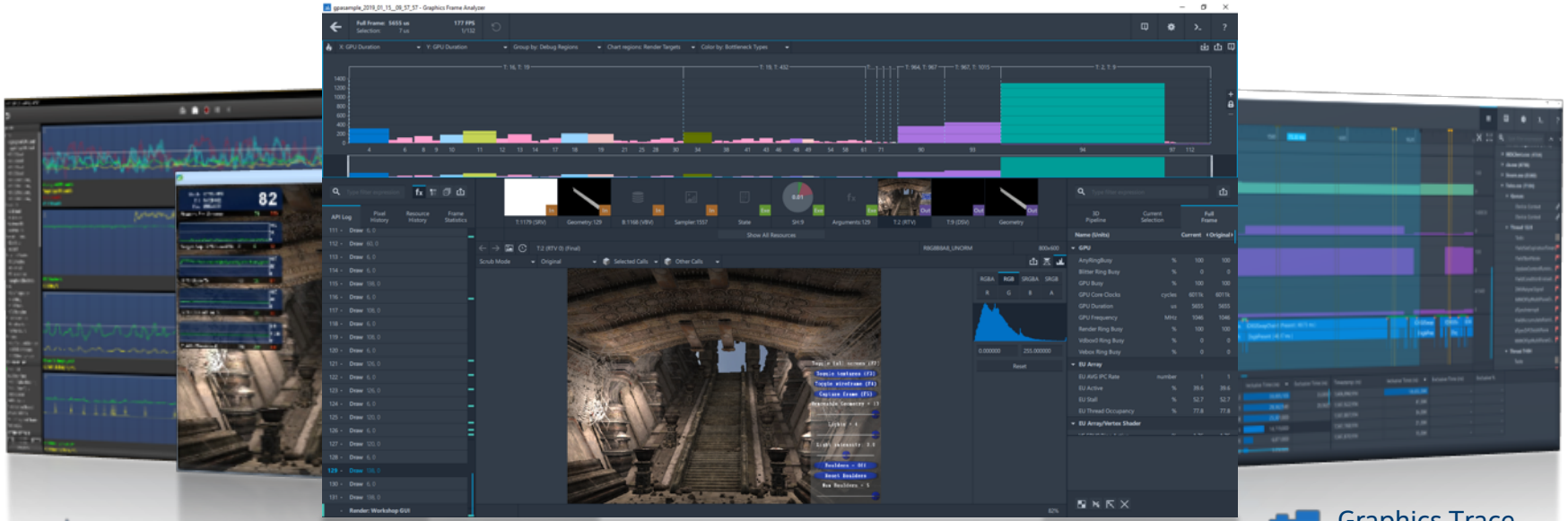


Scaling Graphics Performance Effectively

My super awesome game



Intel® Graphics Performance Analyzers
(Intel GPA)



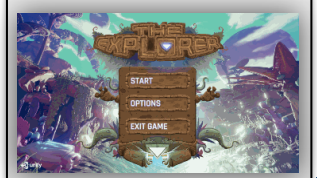
System Analyzer

Graphics Frame Analyzer

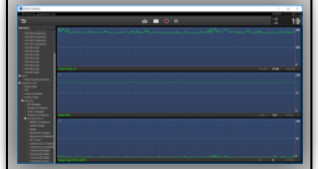
Graphics Trace Analyzer

Profiling Workflow

Identify Scene

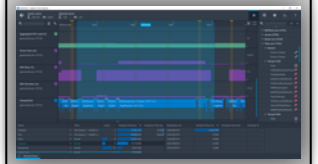


+



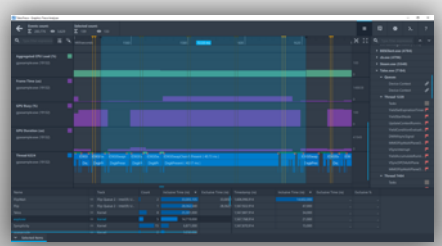
CPU vs GPU bound?

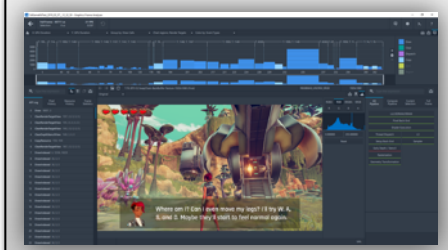
+



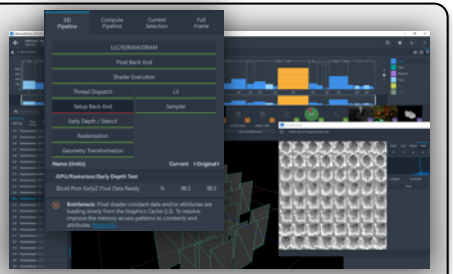
CPU bound

Identify Hotspots via instrumentation





Frame Analysis

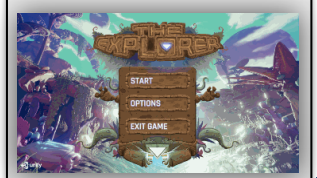


Identify Hotspots

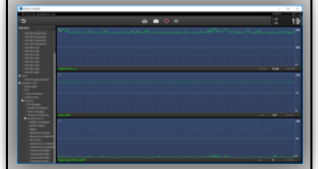
GPU bound

Profiling Workflow

Identify Scene

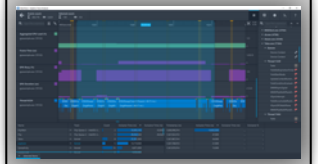


+



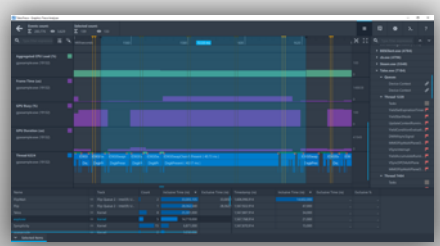
CPU vs GPU bound?

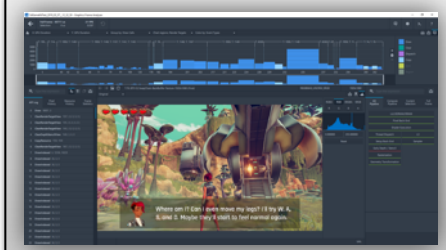
+



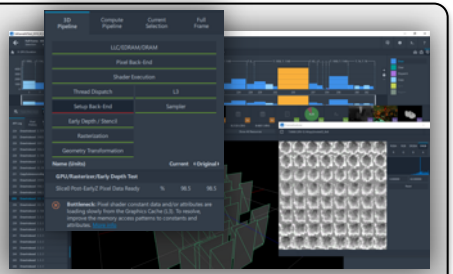
CPU bound

Identify Hotspots via instrumentation





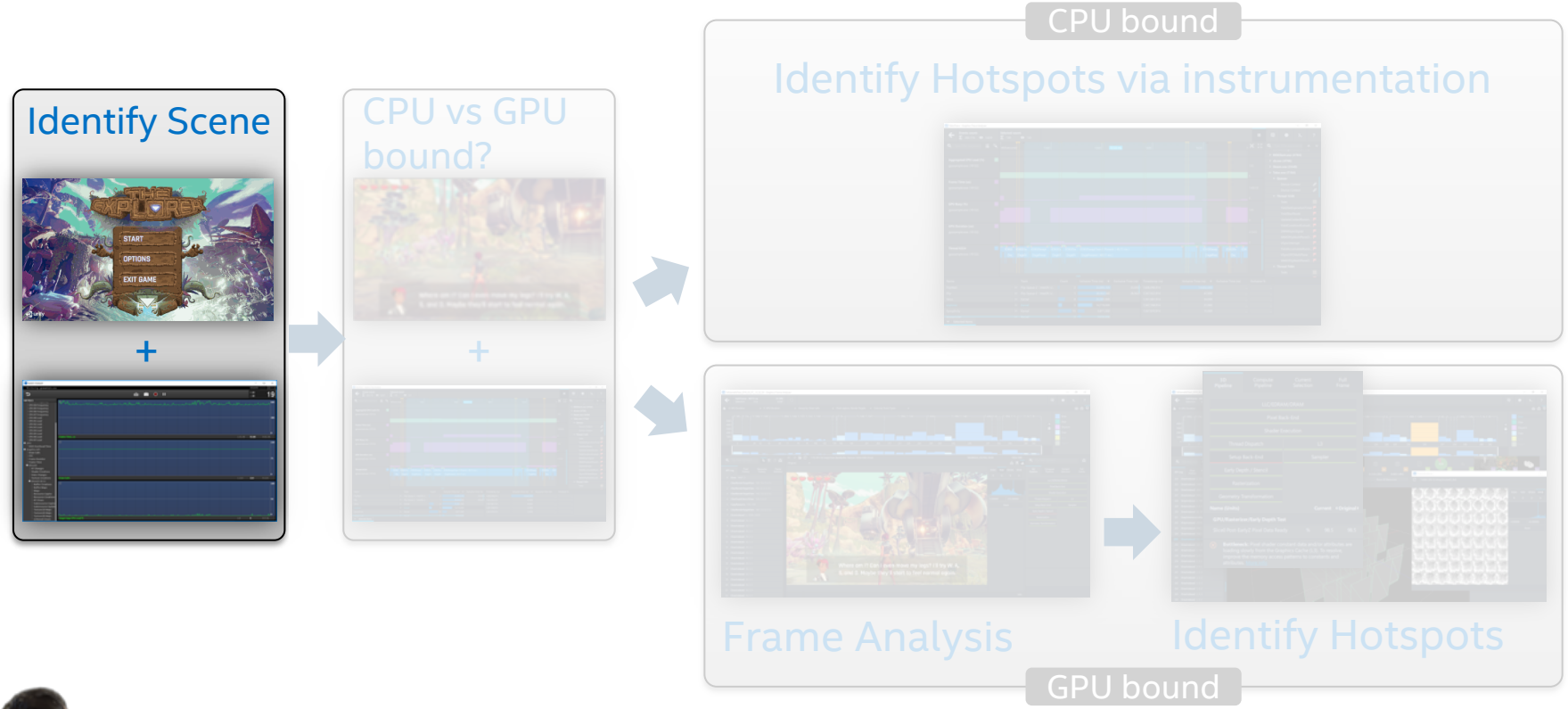
Frame Analysis



Identify Hotspots

GPU bound

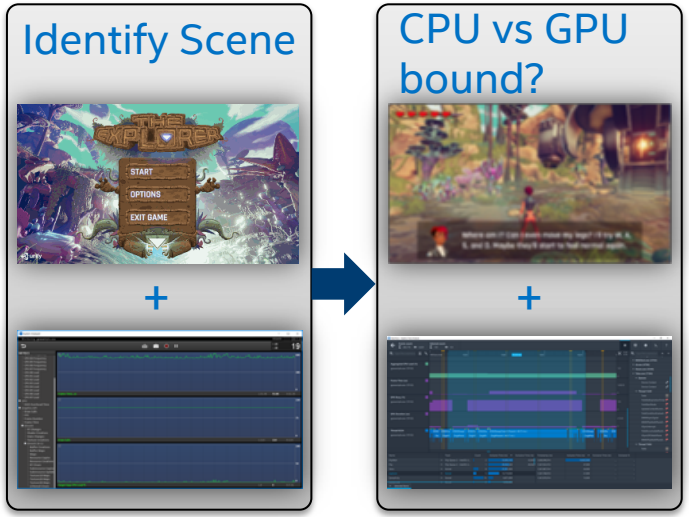
Profiling Workflow



System Analyzer Overview

In Application Live Analysis

Profiling Workflow



CPU bound

Identify Hotspots via instrumentation

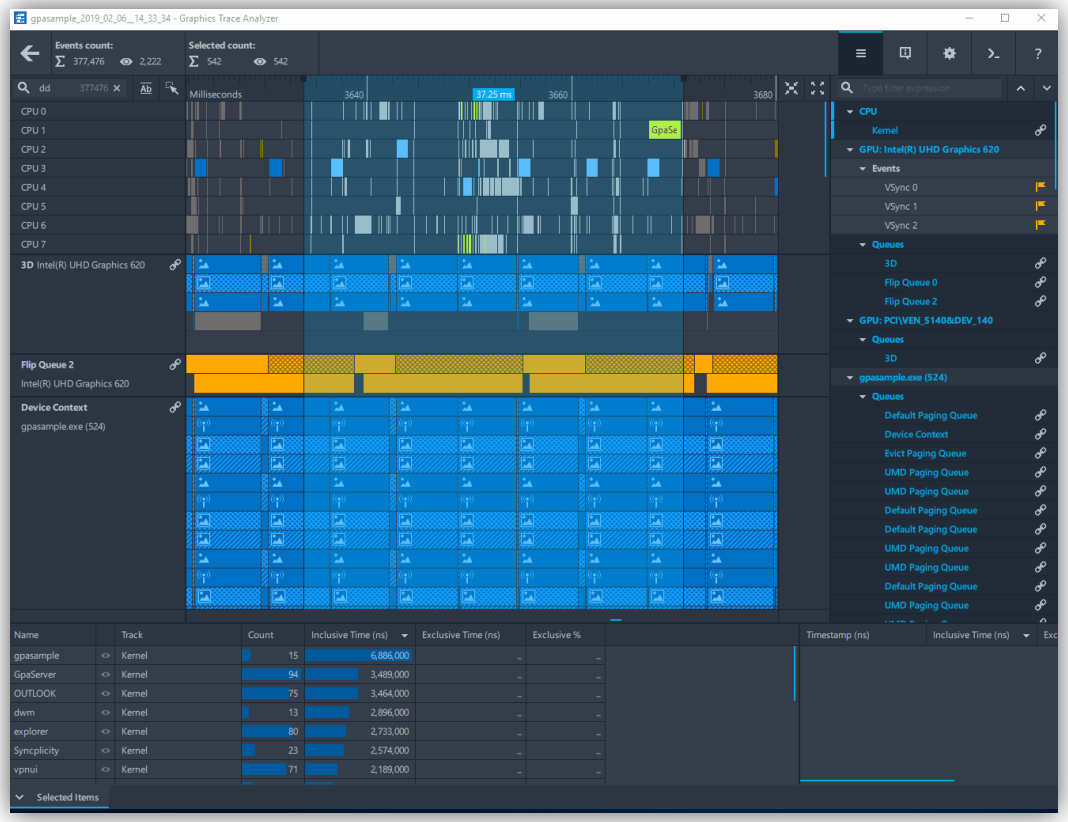
Frame Analysis

Identify Hotspots

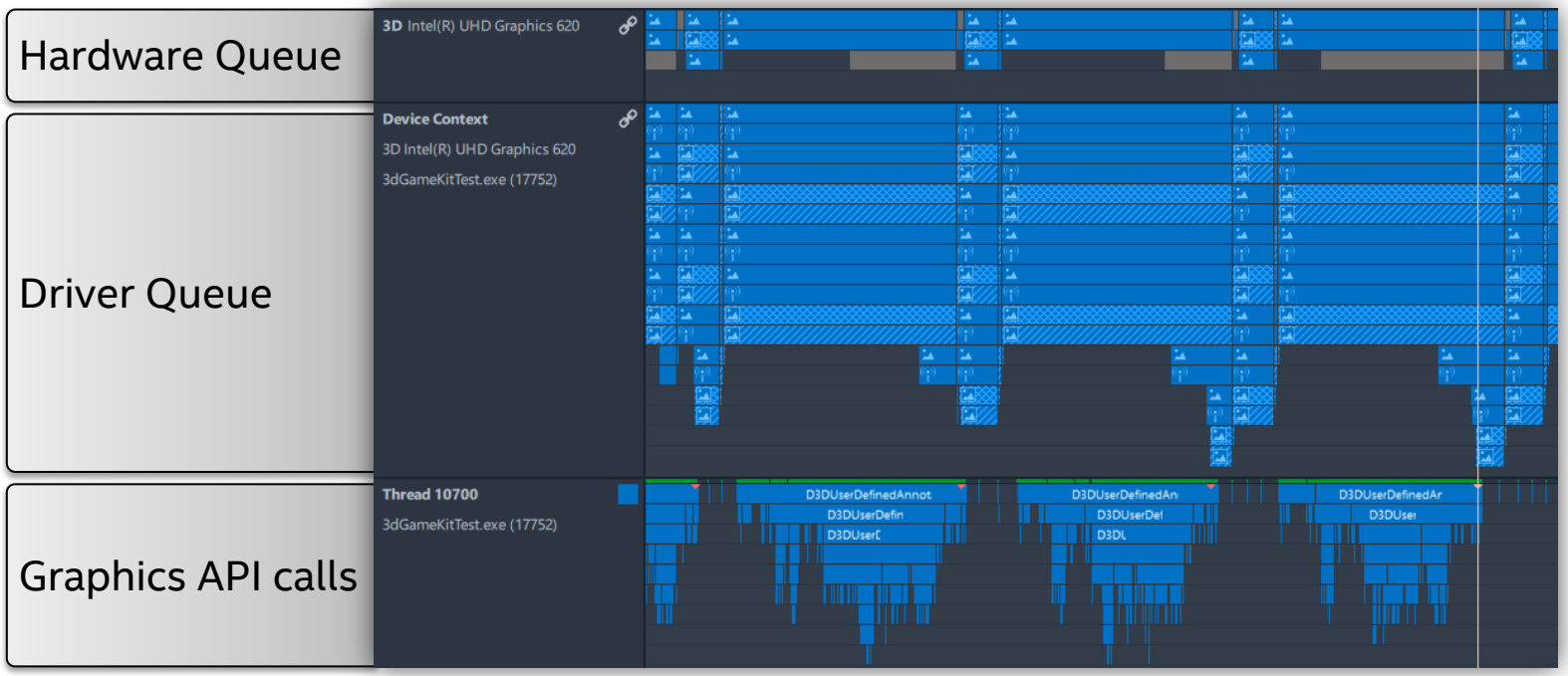
GPU bound

Graphics Trace Analyzer

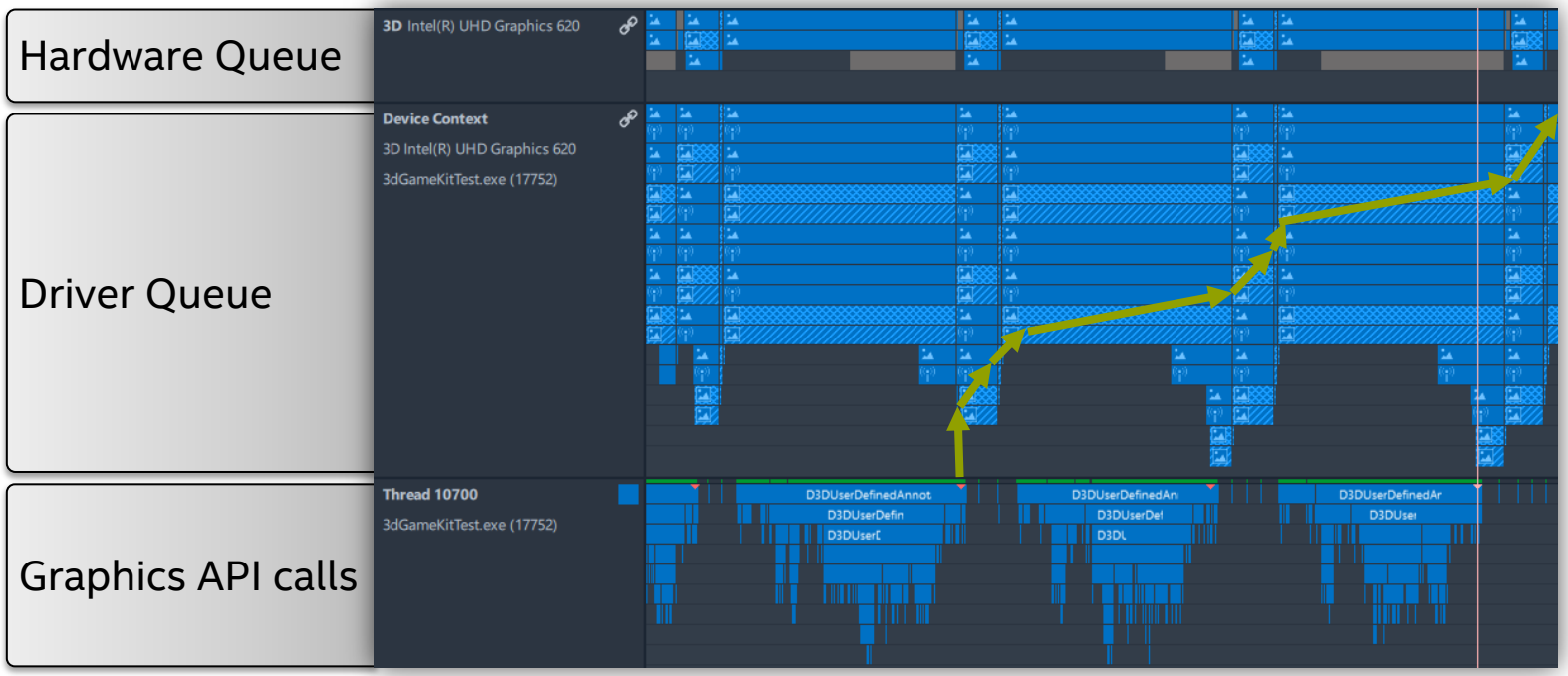
- New UI with improved collection mechanism
- Collect data from various sources correlated in one timeline
  - ETW events
  - Graphics API calls
  - ITT user markers
  - Sampled metrics



Lifetime of Graphics Commands



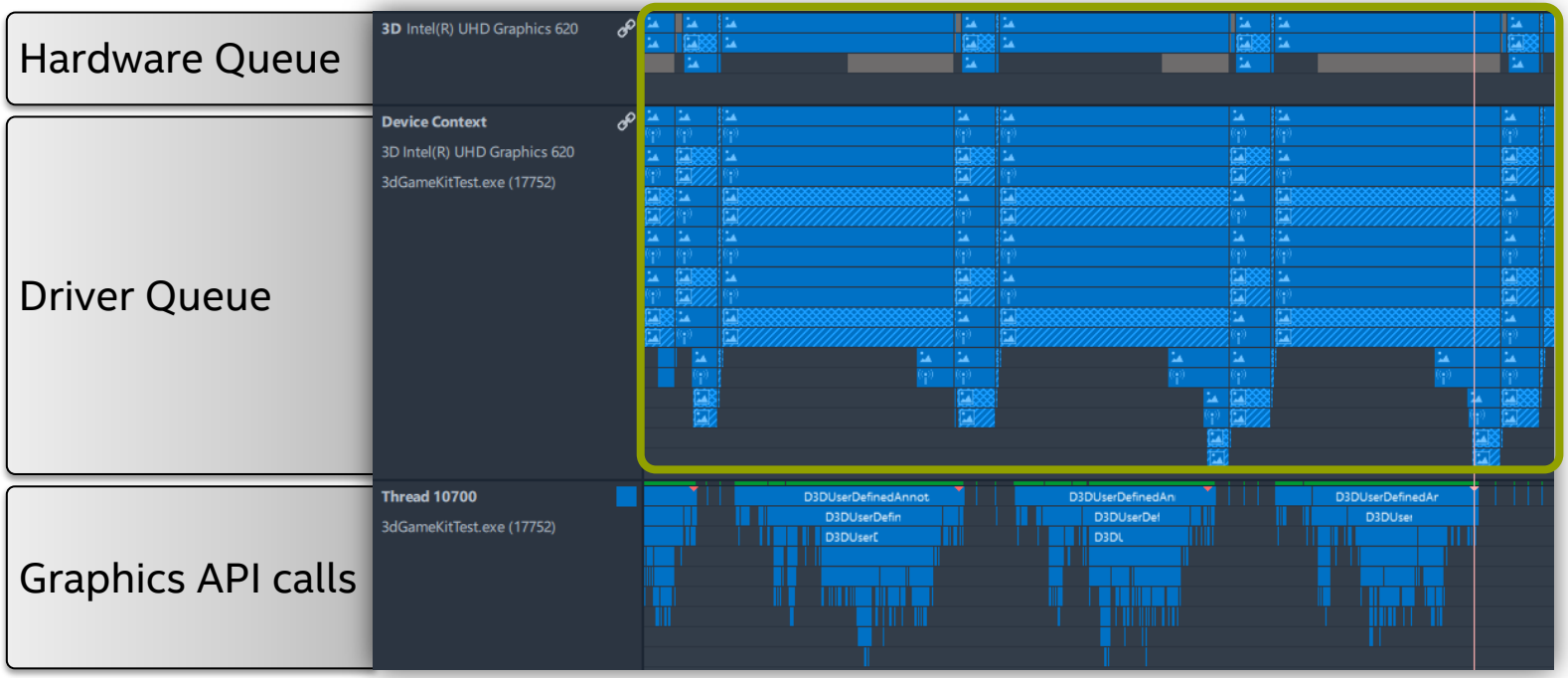
Lifetime of Graphics Commands



Present call in queue until finally executed by GPU

Lifetime of Graphics Commands

No gaps in queue

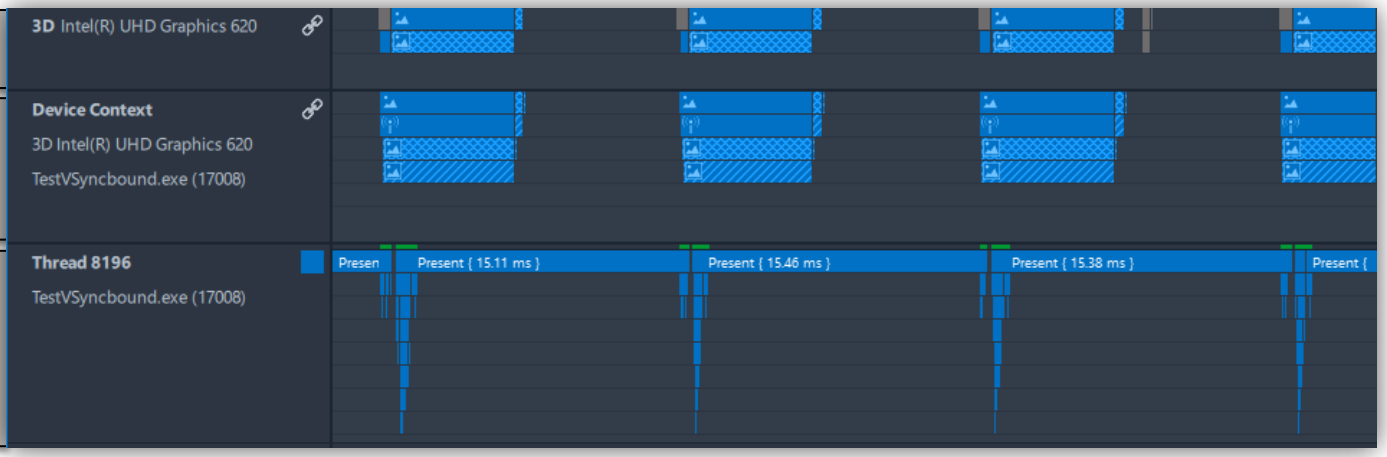


Not GPU bound

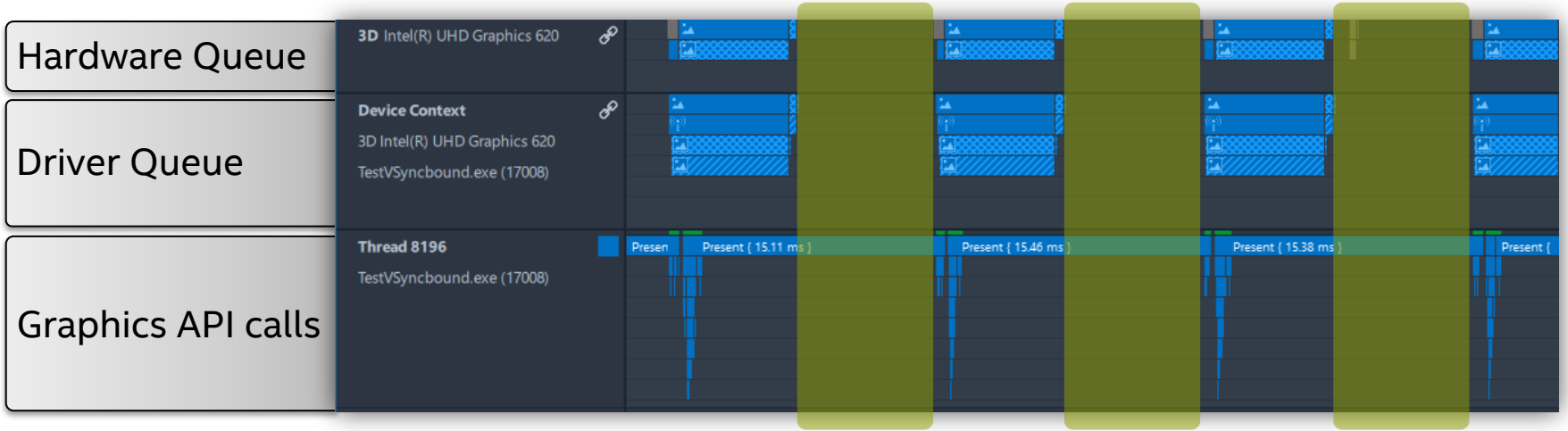
Hardware Queue

Driver Queue

Graphics API calls

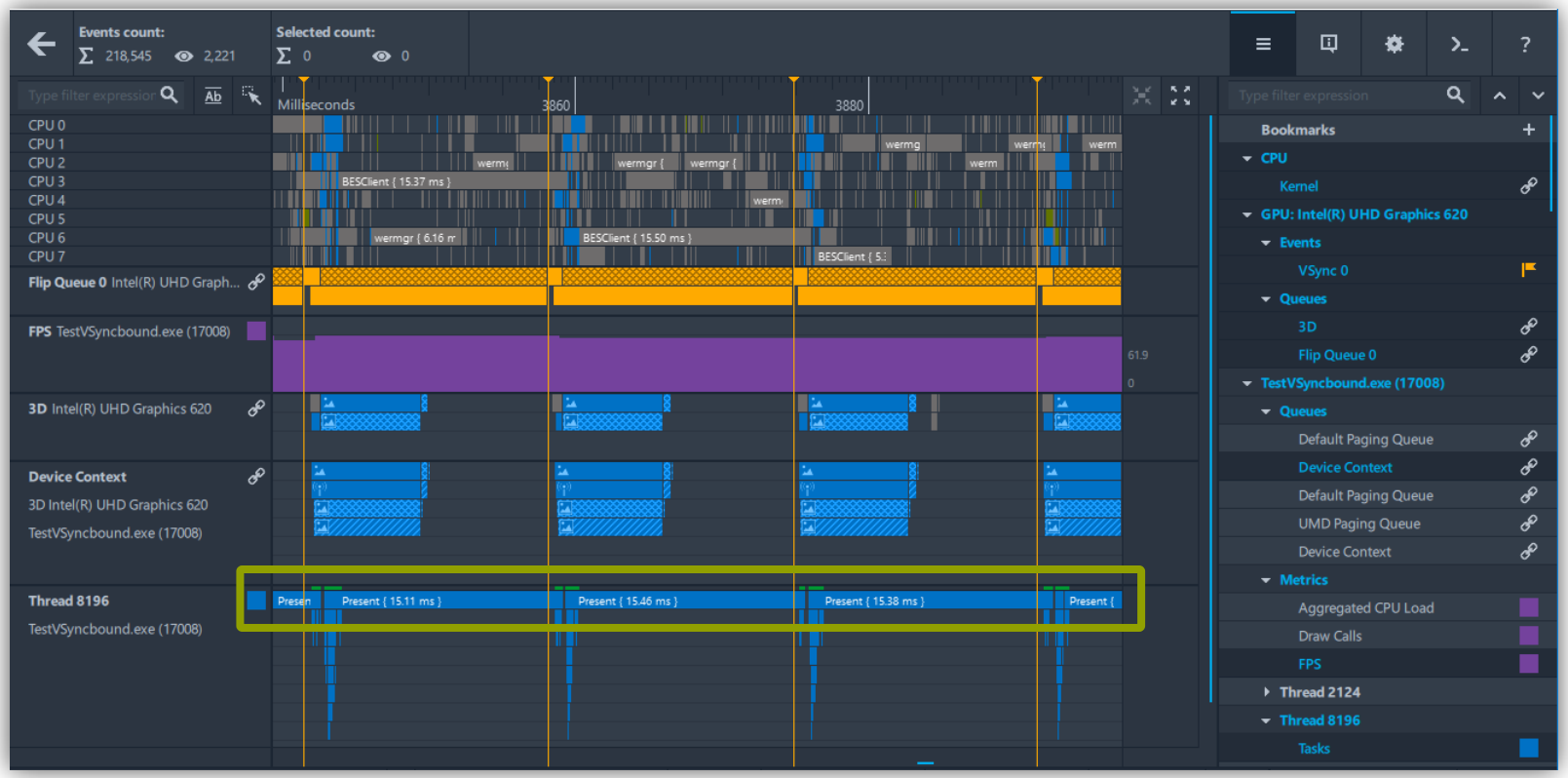


Not GPU bound

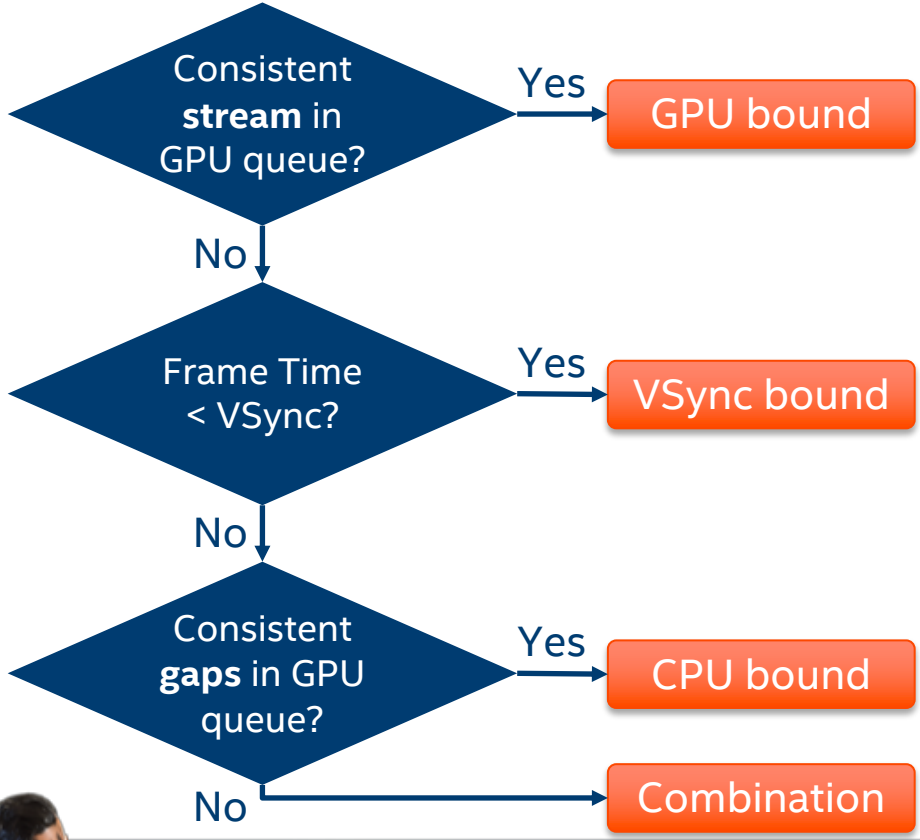


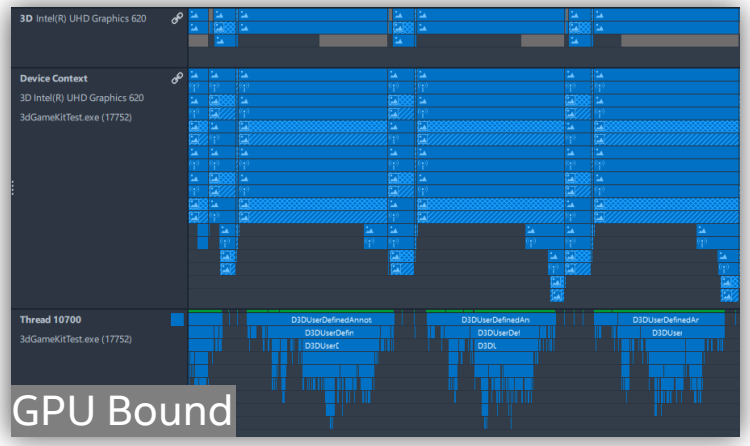
Gaps in queue

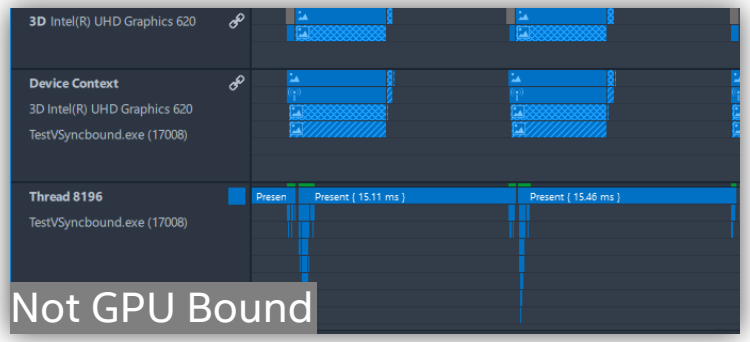
VSync Bound


Events count:
218,545
2,221
Selected count:
0
0
Milliseconds
3860
3880
CPU 0
CPU 1
CPU 2
CPU 3
CPU 4
CPU 5
CPU 6
CPU 7
BESClient { 15.37 ms }
wermgr { 6.16 m
BESClient { 15.50 ms }
Flip Queue 0 Intel(R) UHD Graph...
FPS TestVSyncbound.exe (17008)
61.9
0
3D Intel(R) UHD Graphics 620
Device Context
3D Intel(R) UHD Graphics 620
TestVSyncbound.exe (17008)
Thread 8196
TestVSyncbound.exe (17008)
Present { 15.11 ms }
Present { 15.46 ms }
Present { 15.38 ms }
Bookmarks
CPU
Kernel
GPU: Intel(R) UHD Graphics 620
Events
VSync 0
Queues
3D
Flip Queue 0
TestVSyncbound.exe (17008)
Queues
Default Paging Queue
Device Context
Default Paging Queue
UMD Paging Queue
Device Context
Metrics
Aggregated CPU Load
Draw Calls
FPS
Thread 2124
Thread 8196
Tasks

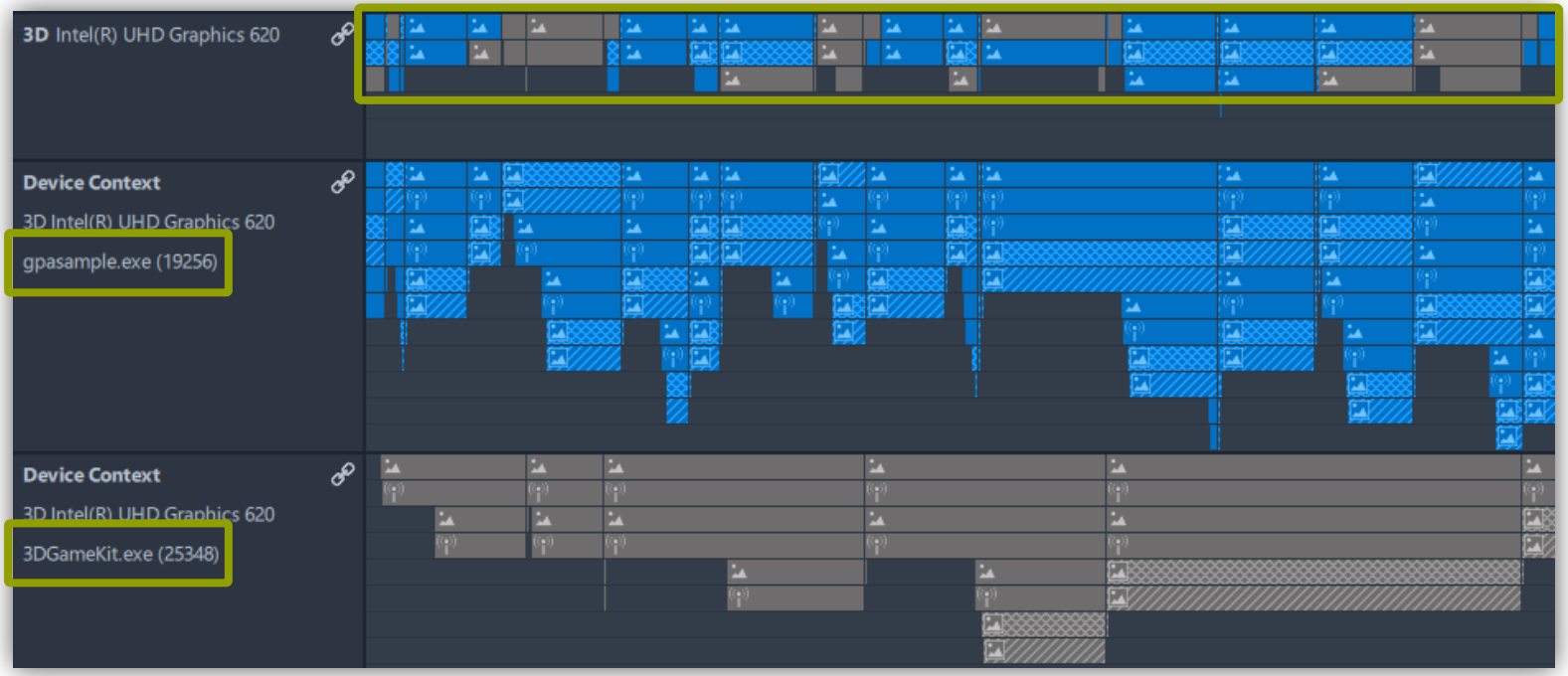
GPU bound vs CPU bound comparison





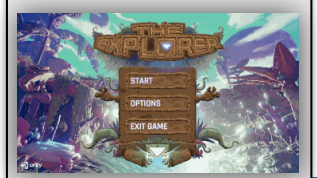


Differentiate GPU Usage per Application

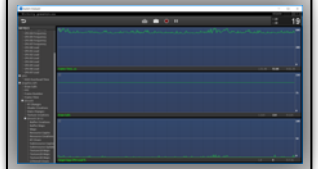


Profiling Workflow

Identify Scene

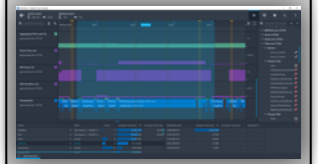


+



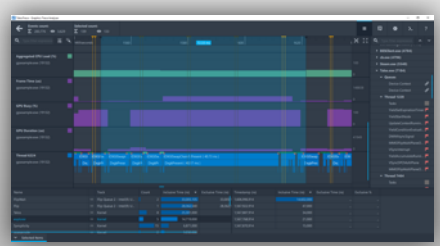
CPU vs GPU bound?

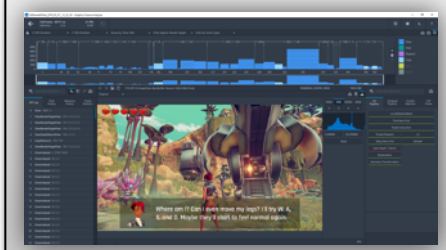
+



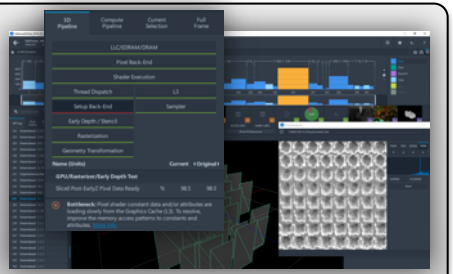
CPU bound

Identify Hotspots via instrumentation





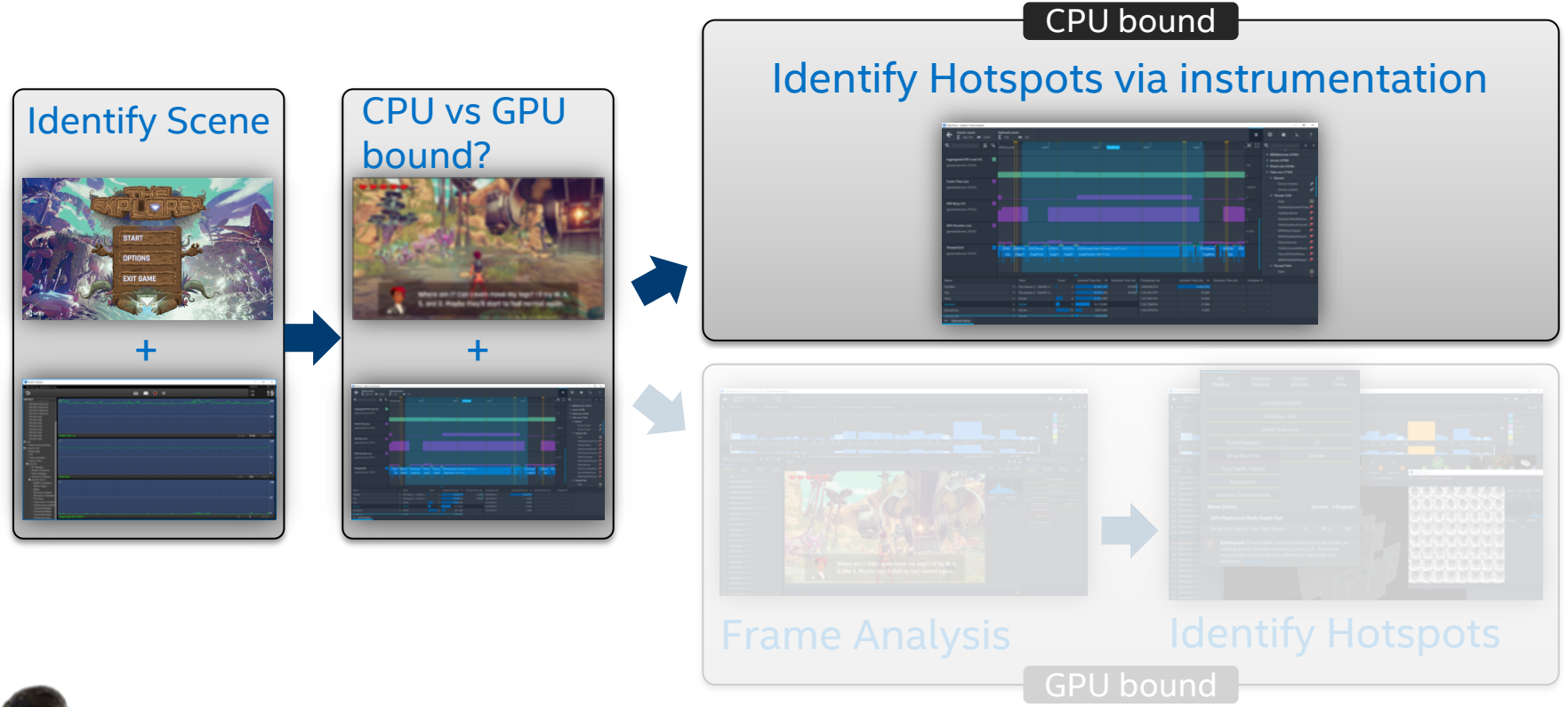
Frame Analysis



Identify Hotspots

GPU bound

CPU Bound


Identify Scene
THE EXPLORER
START
OPTIONS
EXIT GAME
+
CPU vs GPU bound?
+
CPU bound
Identify Hotspots via instrumentation
Frame Analysis
Identify Hotspots
GPU bound

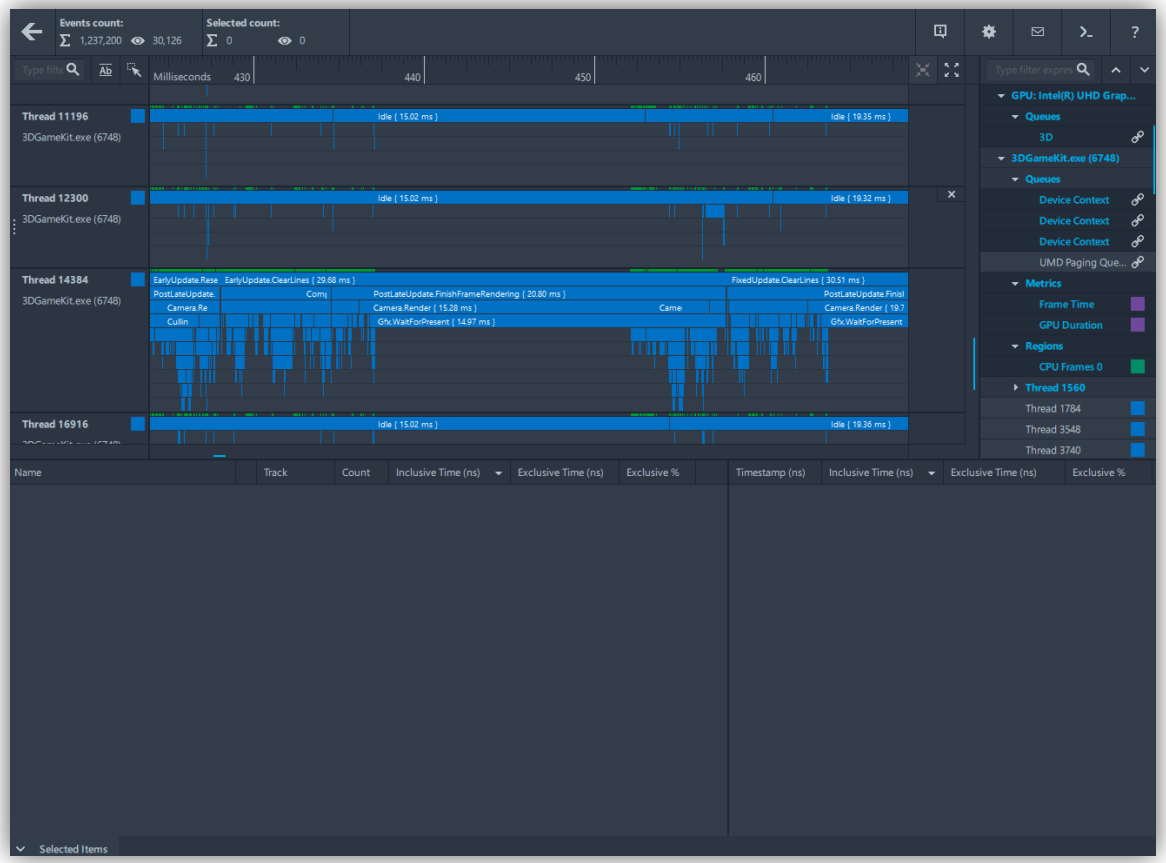
Instrumentation

Instrumentation Alternatives

- ITT
- ETW
- PIX/Graphics API markers

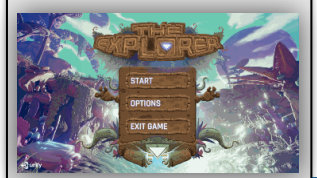
Hotspots from selected region

Dive into region occurrences

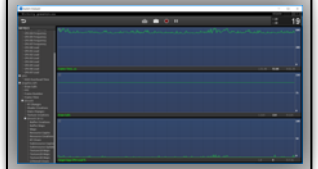


Profiling Workflow

Identify Scene

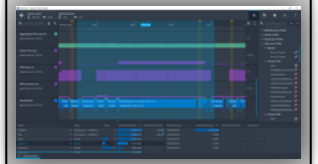


+



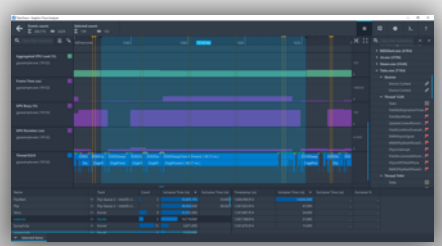
CPU vs GPU bound?

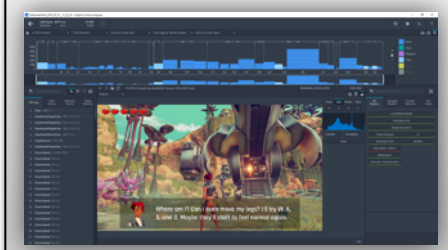
+



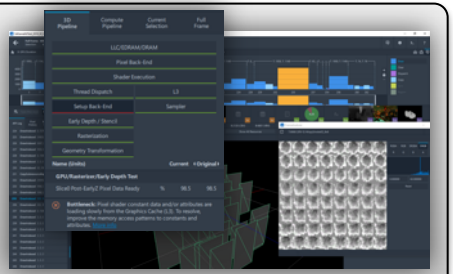
CPU bound

Identify Hotspots via instrumentation





Frame Analysis

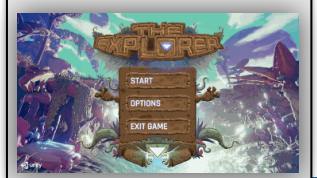


Identify Hotspots

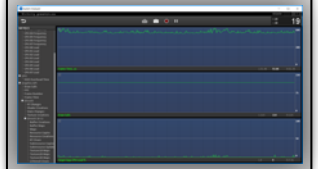
GPU bound

GPU Bound

Identify Scene

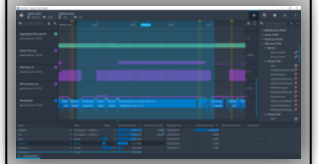


+



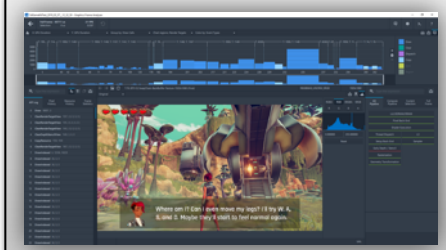
CPU vs GPU bound?

+

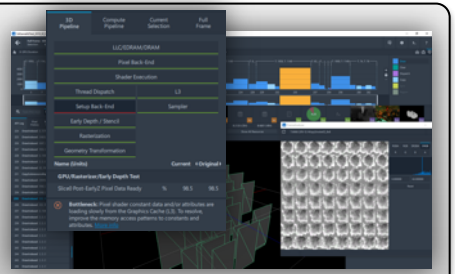


CPU bound

Identify Hotspots via instrumentation



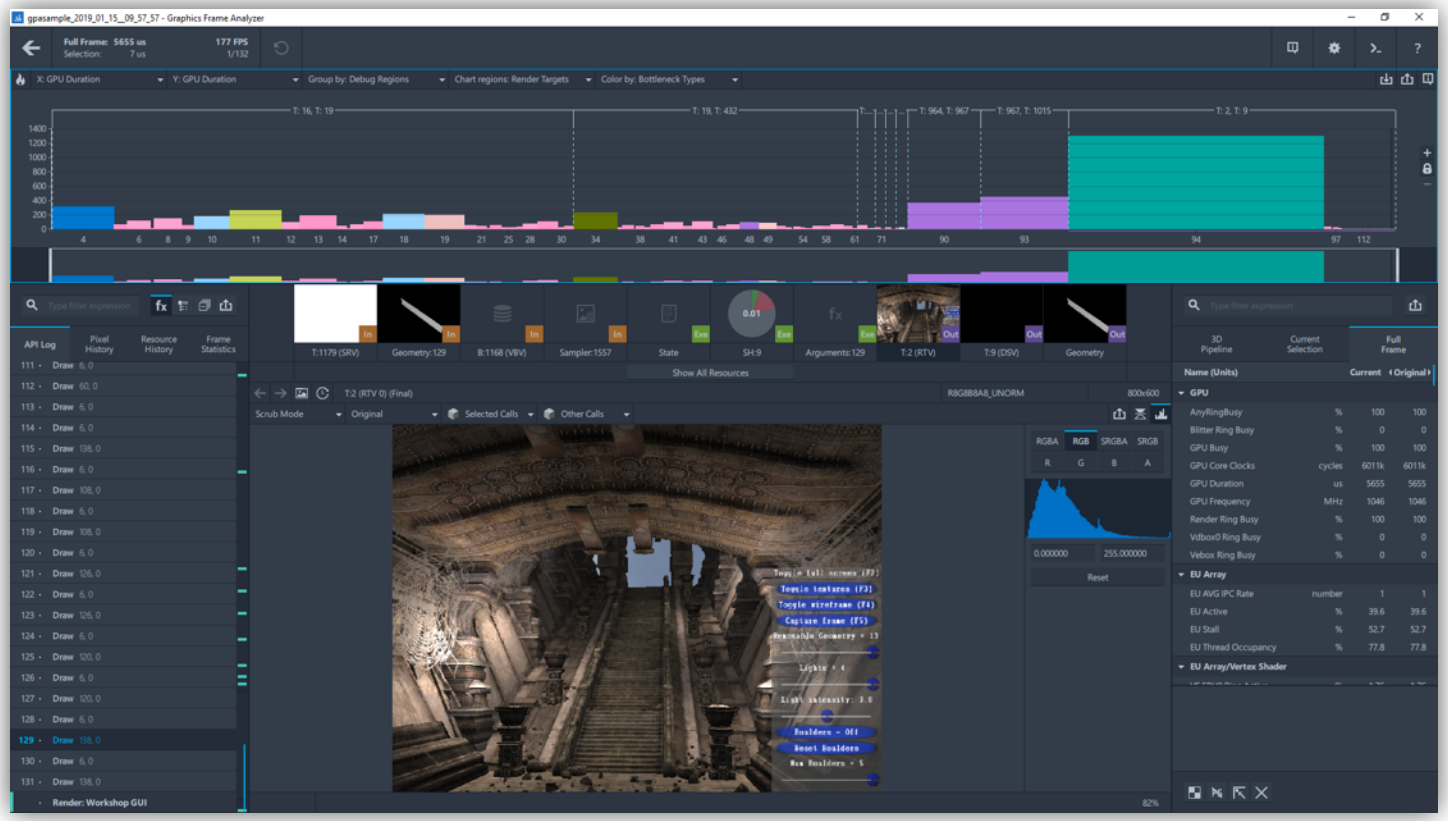
Frame Analysis



Identify Hotspots

GPU bound

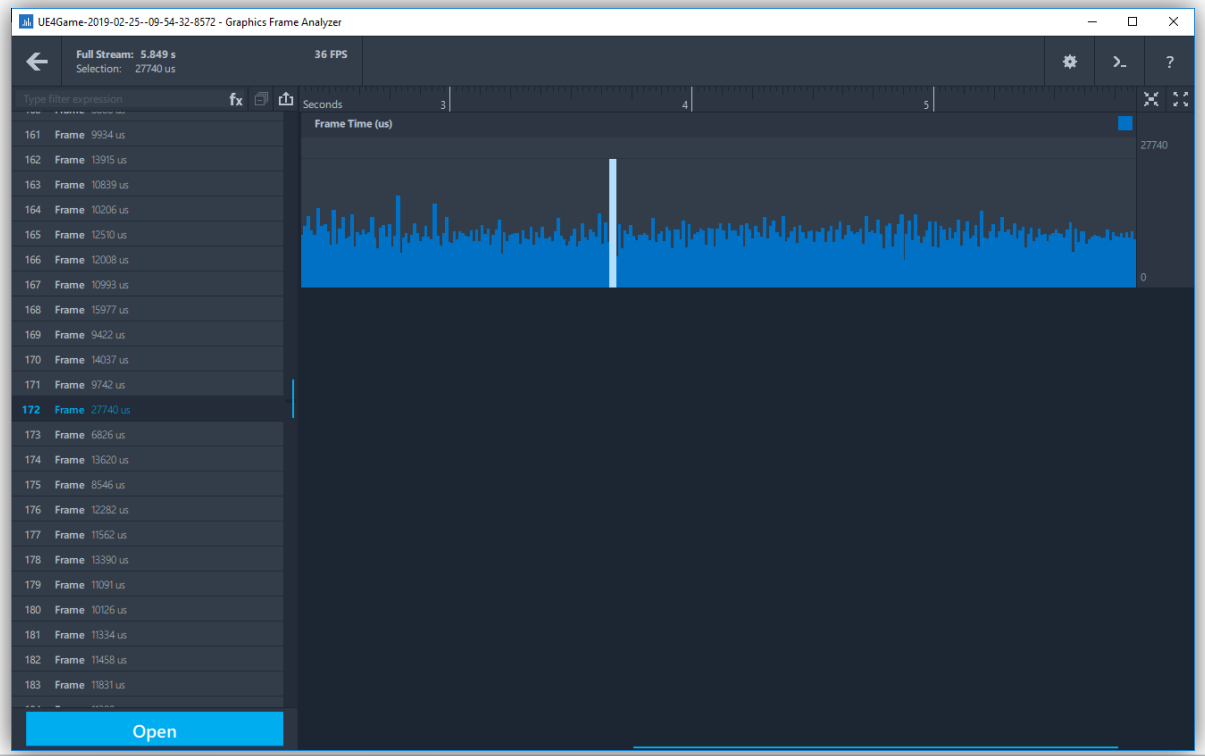
Graphics Frame Analyzer



Multiframe Capture Technical Preview

New in Intel® GPA 2019 R1 Release

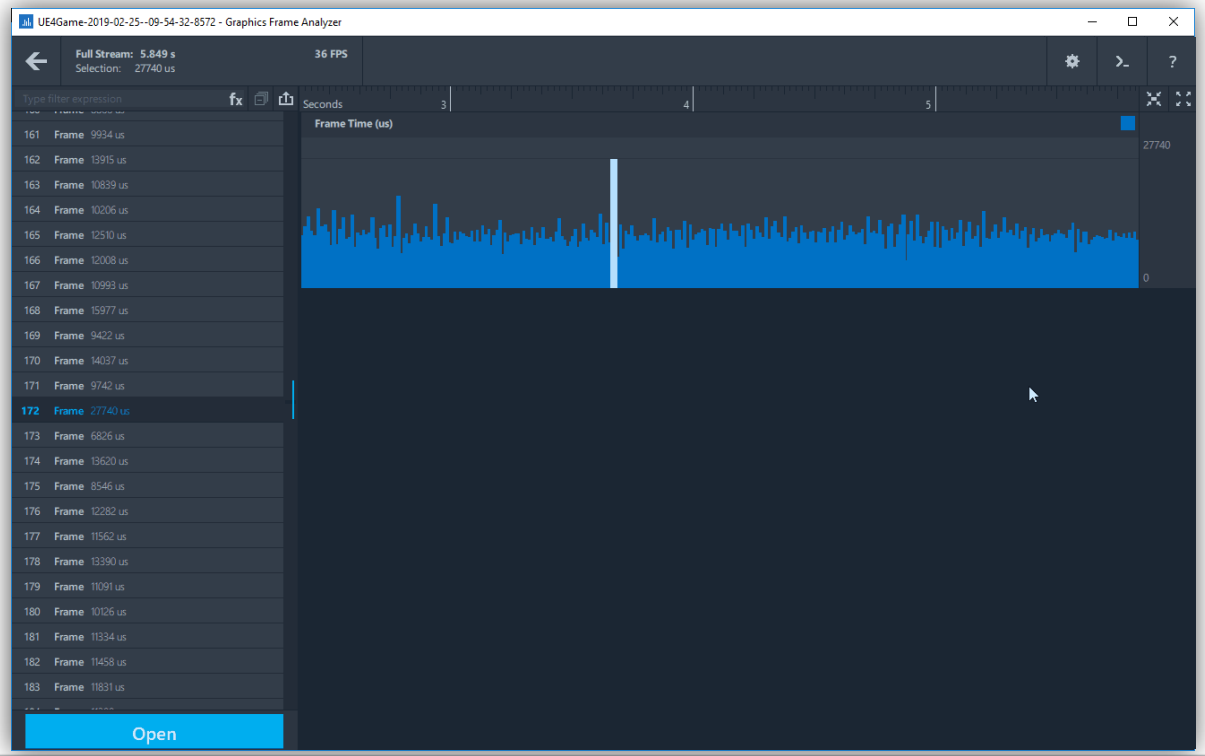
- Difficult problems for single frame analysis
- Debug intermittent glitches and frame hitches
- Profile multi-frame algorithms
- Single frame capture workflow still available



Multiframe Capture Technical Preview

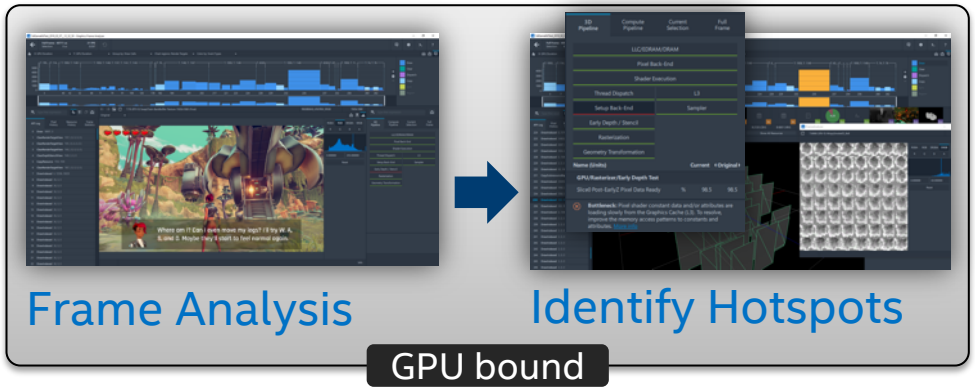
New in Intel® GPA 2019 R1 Release

- Difficult problems for single frame analysis
- Debug intermittent glitches and frame hitches
- Profile multi-frame algorithms
- Single frame capture workflow still available



GPU Bound

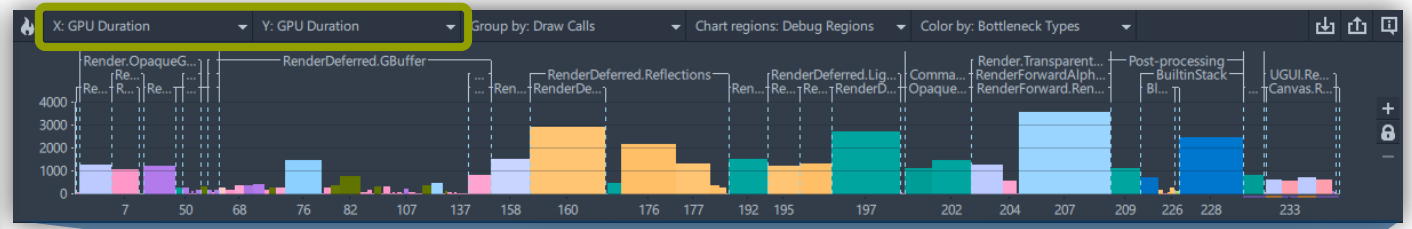
- Frame Analysis to identify most expensive calls
- Easy hotspot analysis for automatic bottleneck root cause analyzer

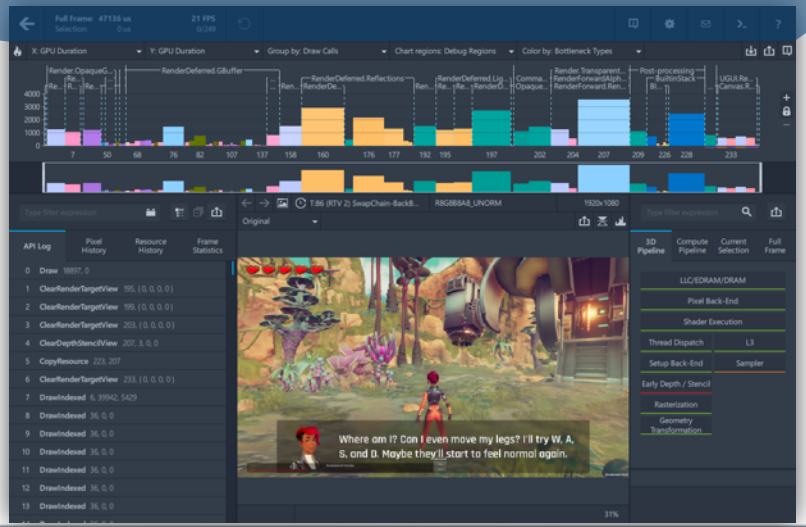


Find most expensive calls

Bottom up approach

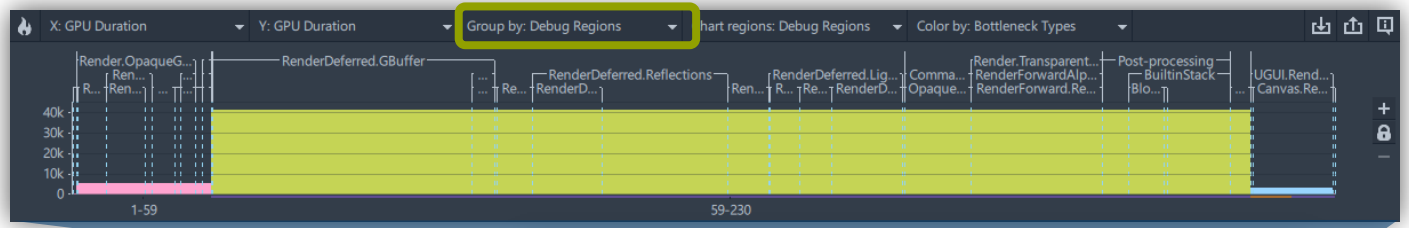
- EU Stall
- EU Thread Occupancy
- Early Depth Bottleneck
- Early Depth Test Fails
- Early Hi-Depth Test Fails
- GPU Busy
- GPU Core Clocks
- GPU Duration
- GPU Frequency
- GS Bottleneck
- GS Invocations
- GS Threads Dispatched
- GTI Depth Throughput
- GTI Fixed Pipe Throughput
- GTI HDC TLB Lookup Throughput

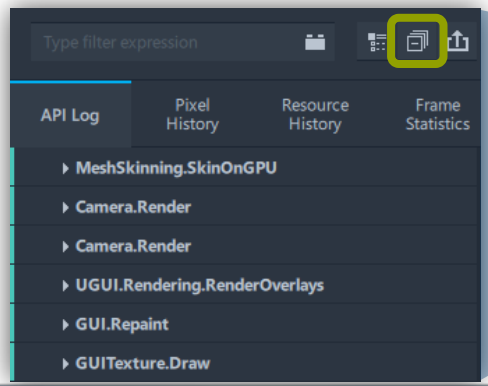


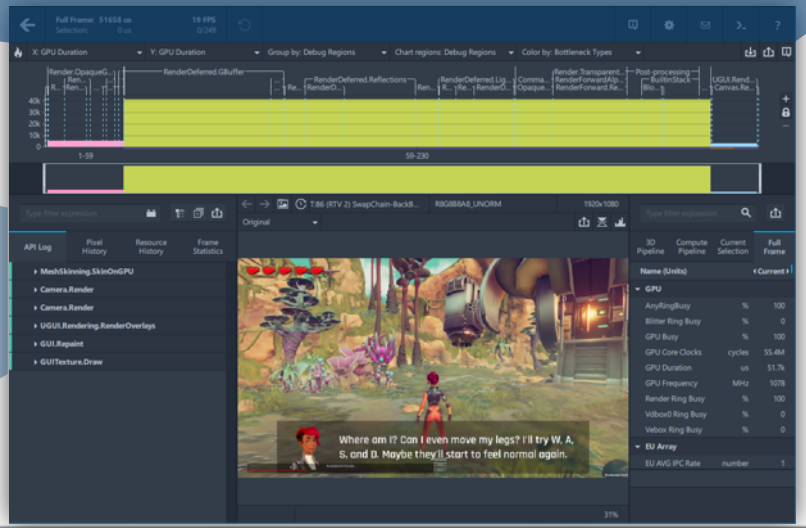


Find most expensive calls

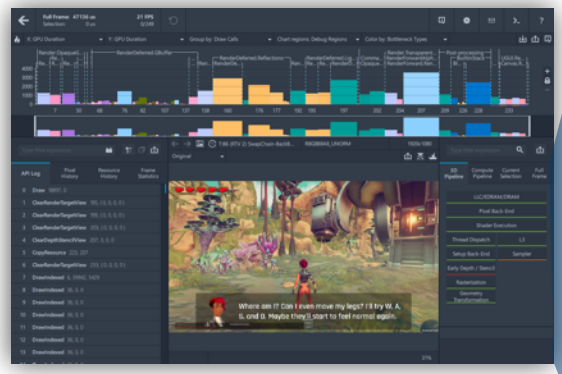
Top down approach

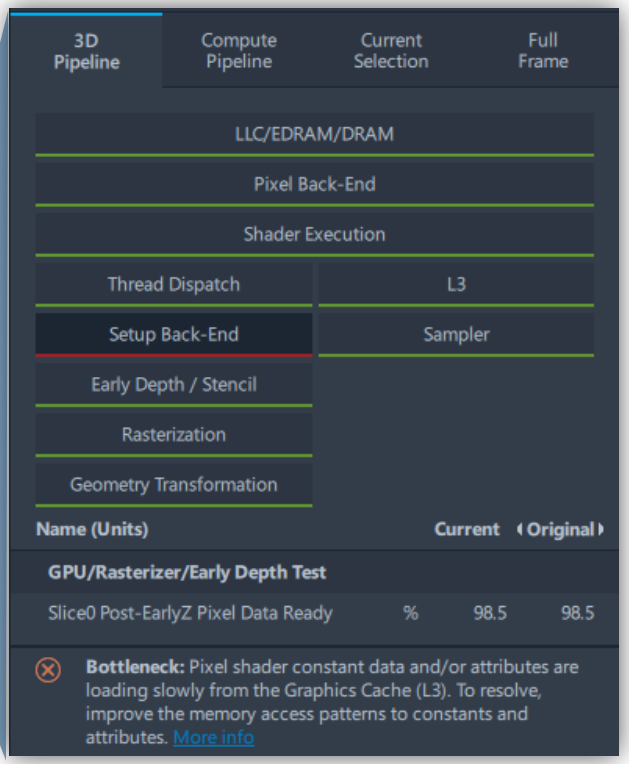


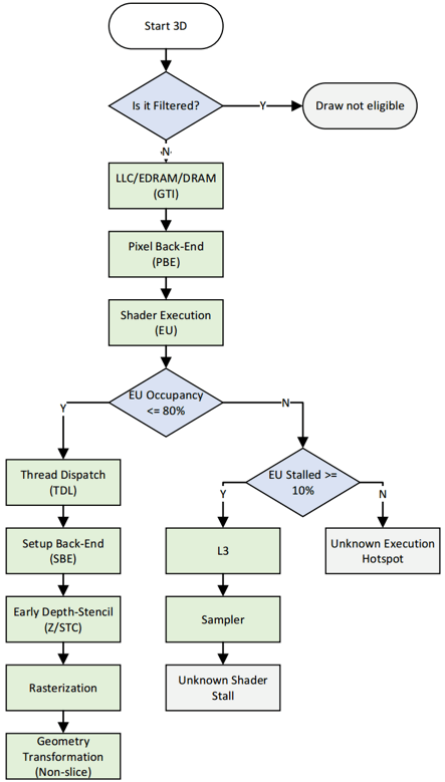




Hotspot Analysis Overview

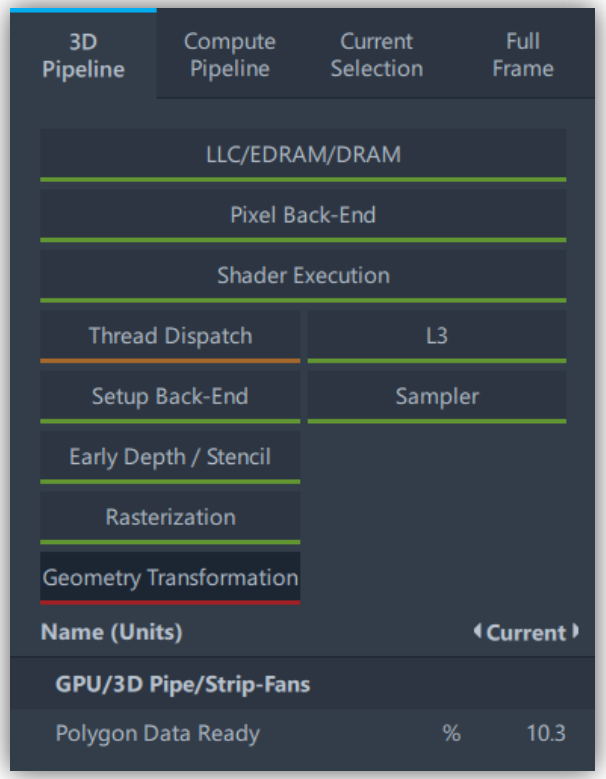




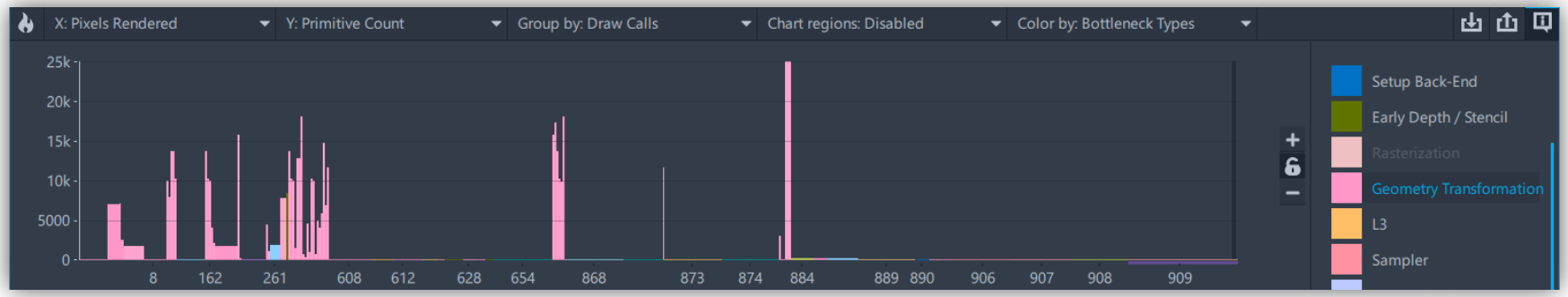


Geometry Transformation

- Bottleneck in Geometry Transformation
- Things to look out for and mitigation:
  - Dense geometry rendered to small area
    - More aggressive LODs or decimation needed
  - Icebergs (5% in view, 95% out)
    - Slice geometry for better culling
  - Stragglers (not in view but not culled)
    - Debug why is object not being culled

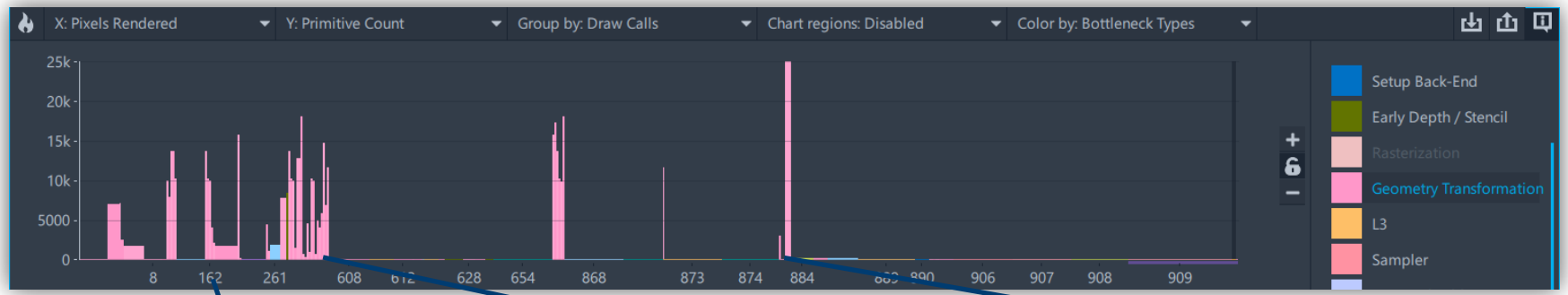


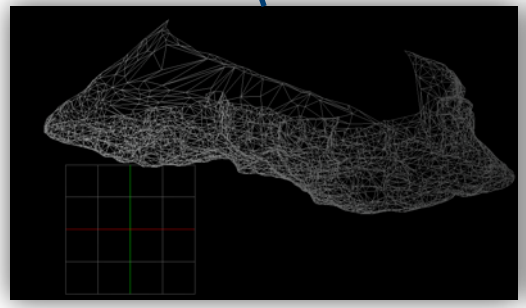
Geometry Transformation



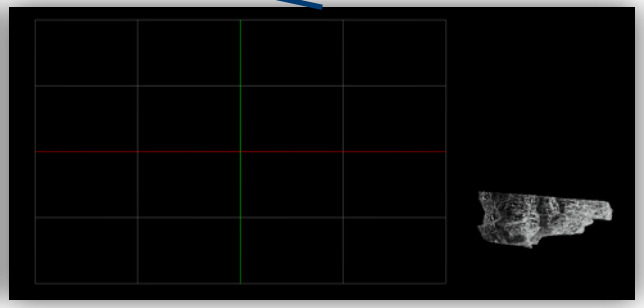
- X Axis: Pixels Rendered
- Y Axis: Primitive Count
- Tall skinny bars, big objects rendered to small amount of pixels

Geometry Transformation

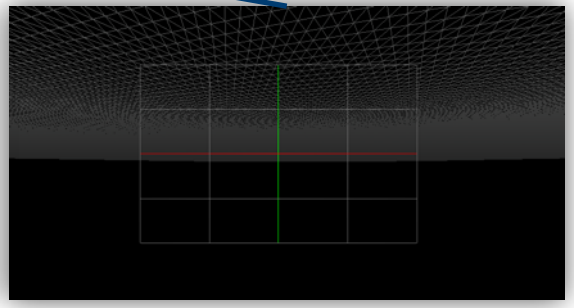




Iceberg

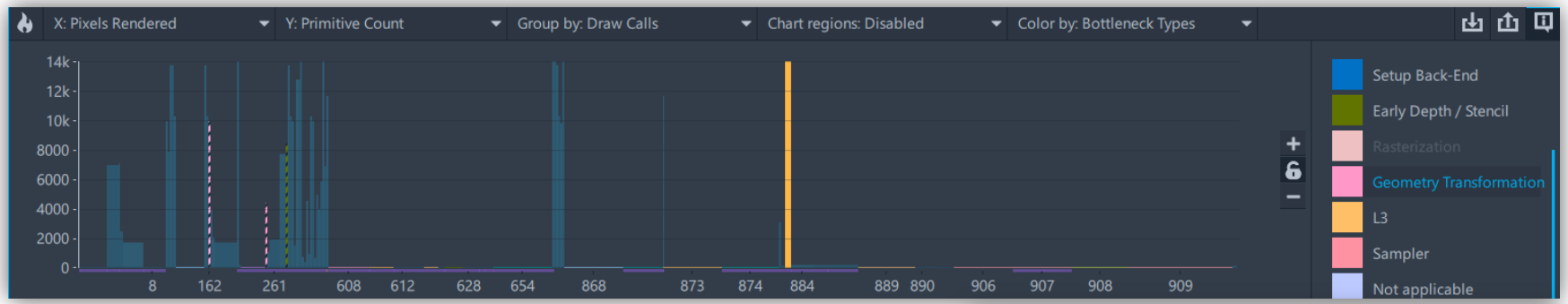


Stragglers

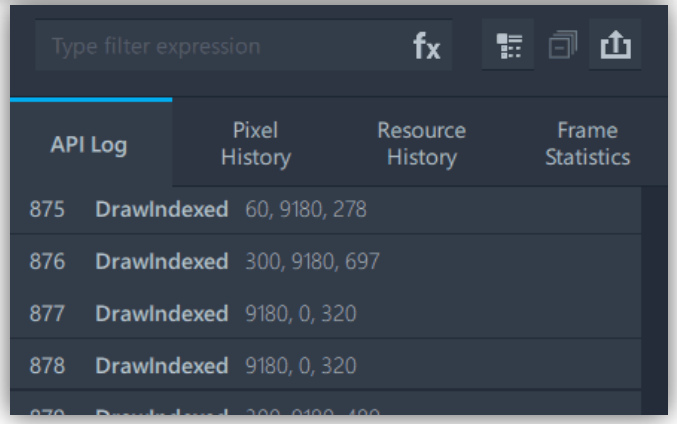


Dense/Iceberg

Geometry Transformation

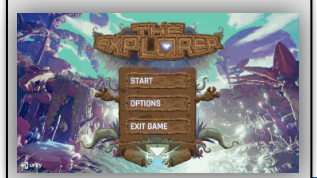


- New Python plugin system
- =metric(“GPU Duration”, “>”, 200)
- Preloaded with several plugins
- Create your own!

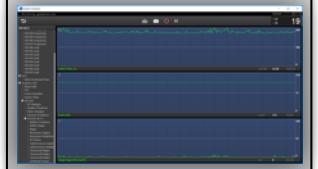


Profiling Workflow

Identify Scene

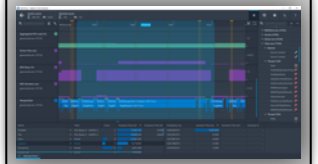


+



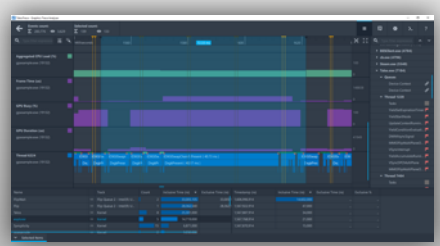
CPU vs GPU bound?

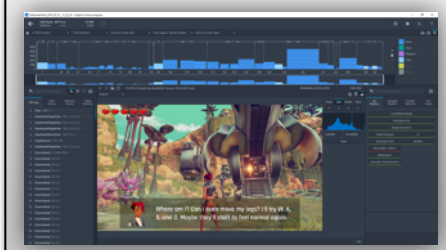
+



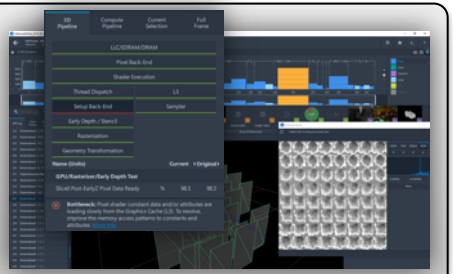
CPU bound

Identify Hotspots via instrumentation





Frame Analysis



Identify Hotspots

GPU bound

Case Study: Unity 3D Game Kit

Overview

Tutorial project in Unity asset store
3 built-in quality settings ‘Performance’, ‘Balanced’ and ‘Fantastic’
Our Goal: ‘Performance’ mode FPS on ‘Fantastic’ mode visual quality

Case Study: Unity 3D Game Kit

Bridge the gap

What makes ‘Fantastic’ look much better than ‘Performance’?

Case Study: Unity 3D Game Kit

Set your performance goals

What makes ‘Fantastic’ look much better than ‘Performance’?

Draw distance

Water reflections

Shadows

Case Study: Unity 3D Game Kit

Spoiler

What makes ‘Fantastic’ look much better than ‘Performance’?

Draw distance

Water reflections

Shadows

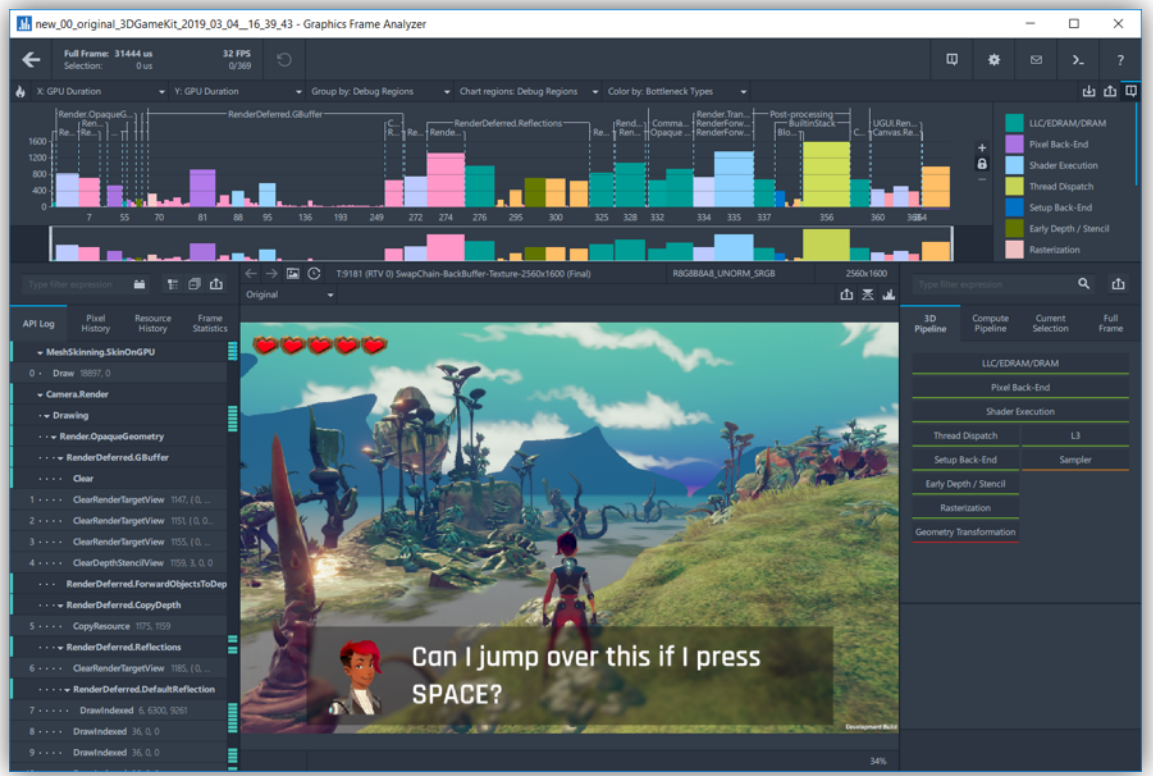
Case Study: Unity 3D Game Kit

How did we get there?

Tug of war, make some room then bring stuff back in

Case Study: Unity 3D Game Kit

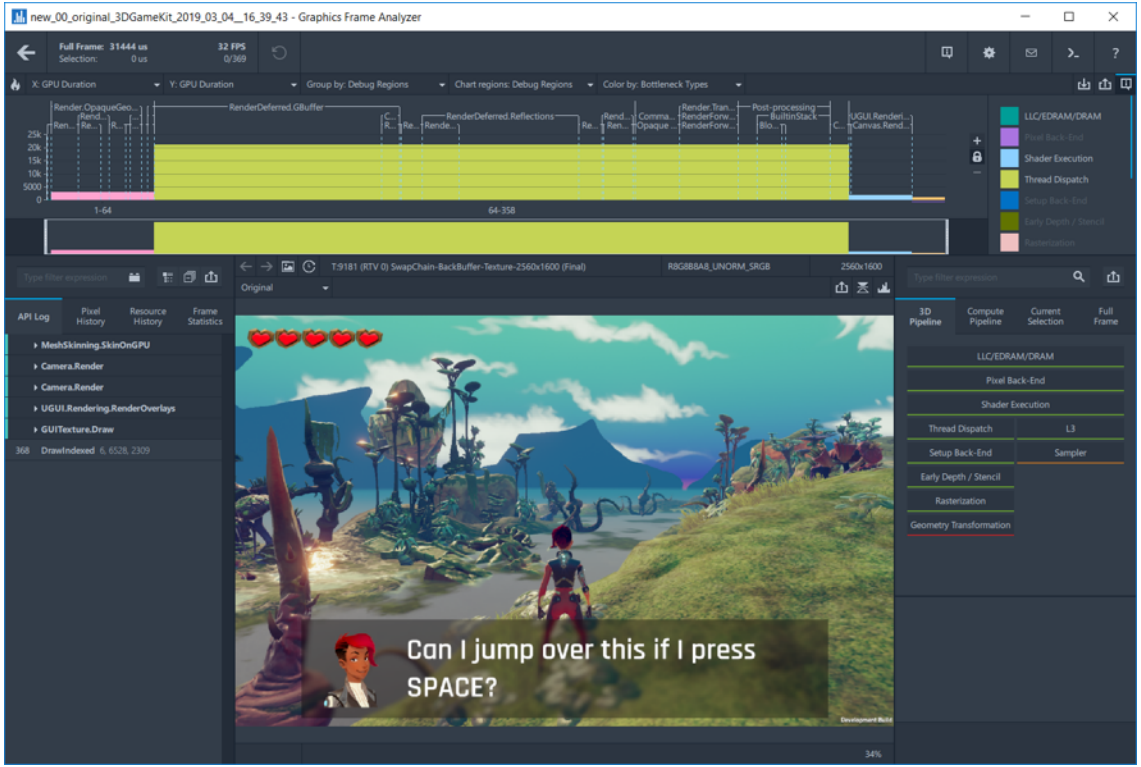
Understand the game's anatomy



Case Study: Unity 3D Game Kit

Understand the game's anatomy

Frame Budget
(30 FPS)

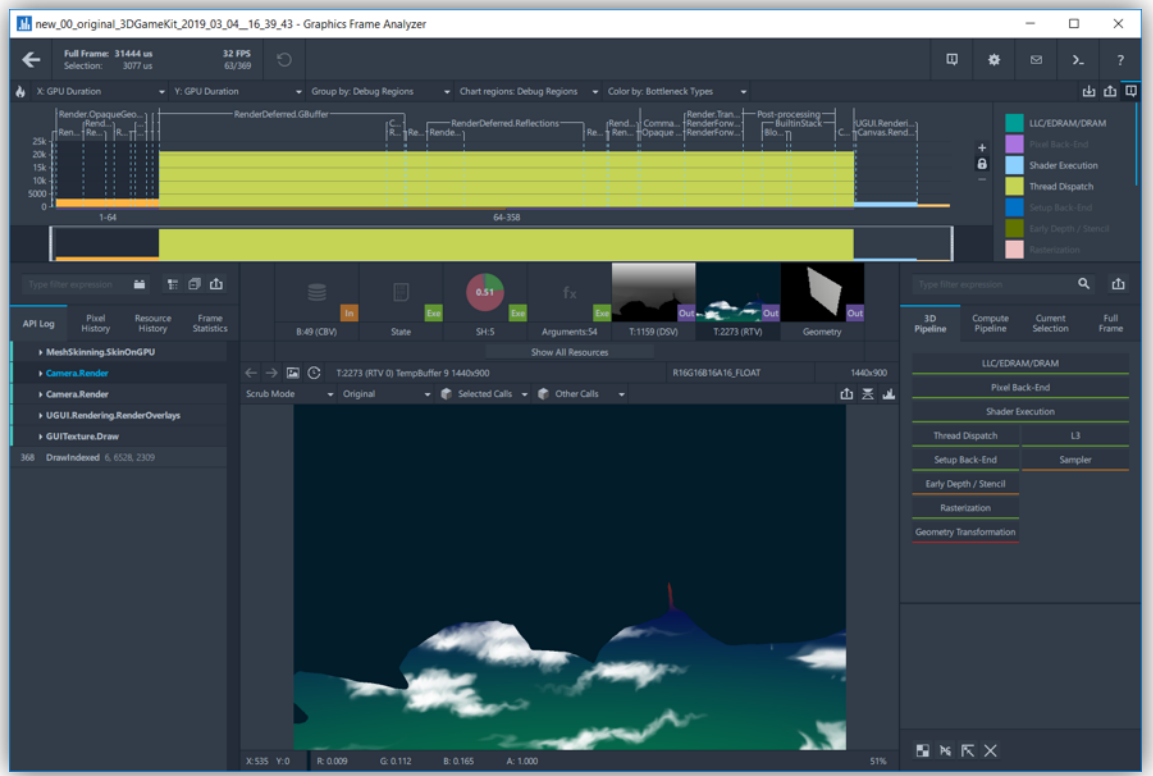


Case Study: Unity 3D Game Kit

Skybox takes 3ms?

~10% of frame budget

Frame Budget
(30 FPS)

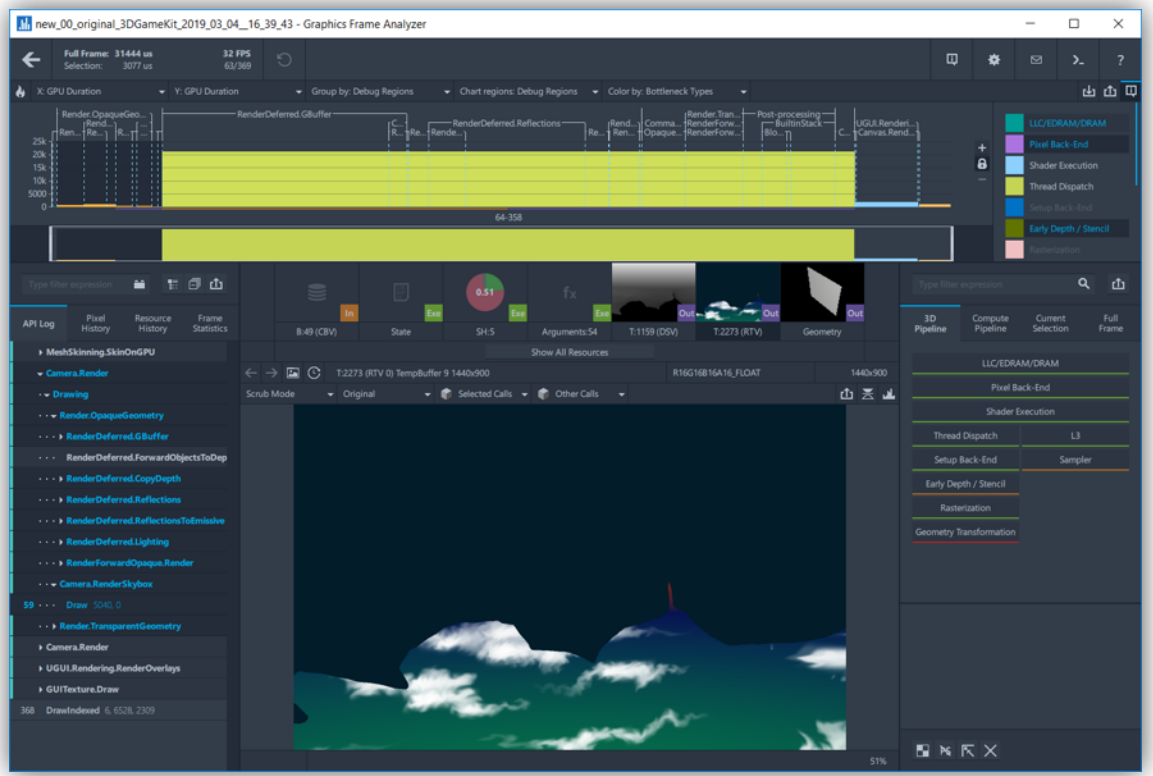


Case Study: Unity 3D Game Kit

Skybox camera set as deferred

~10% of frame budget

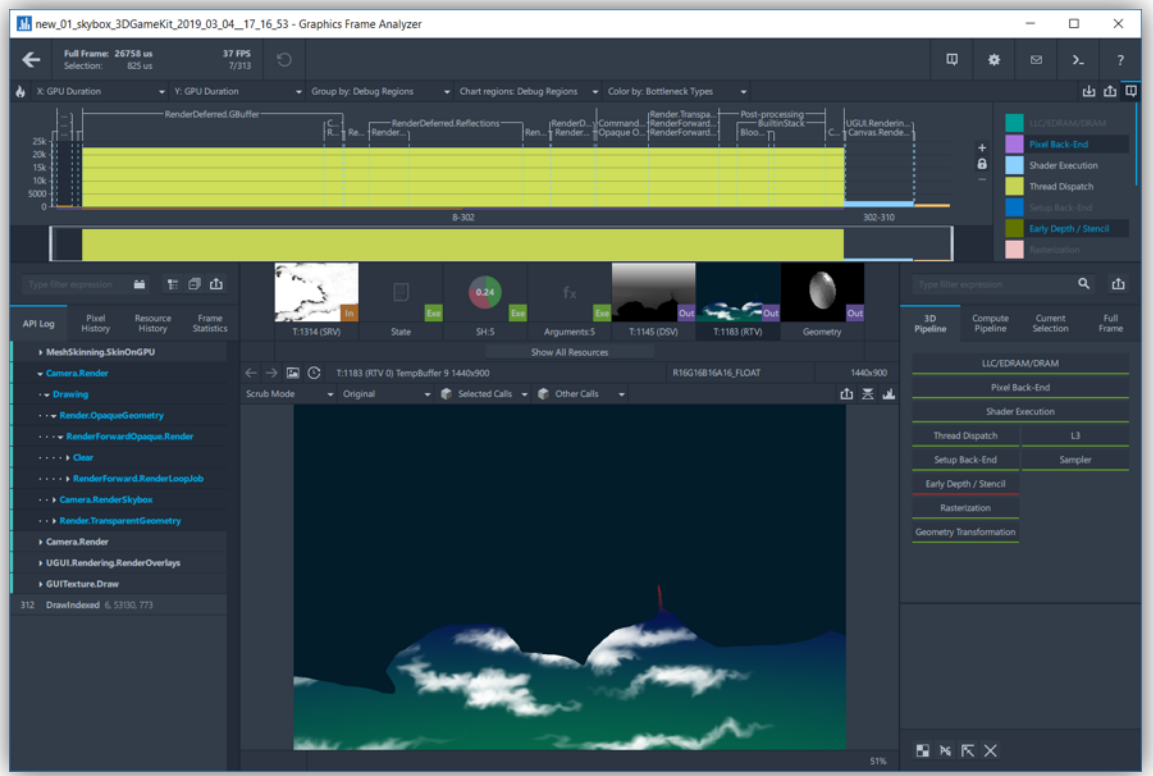
Frame Budget
(30 FPS)



Case Study: Unity 3D Game Kit

Skybox camera set as forward 0.8ms 2.4% of frame budget down from 10%

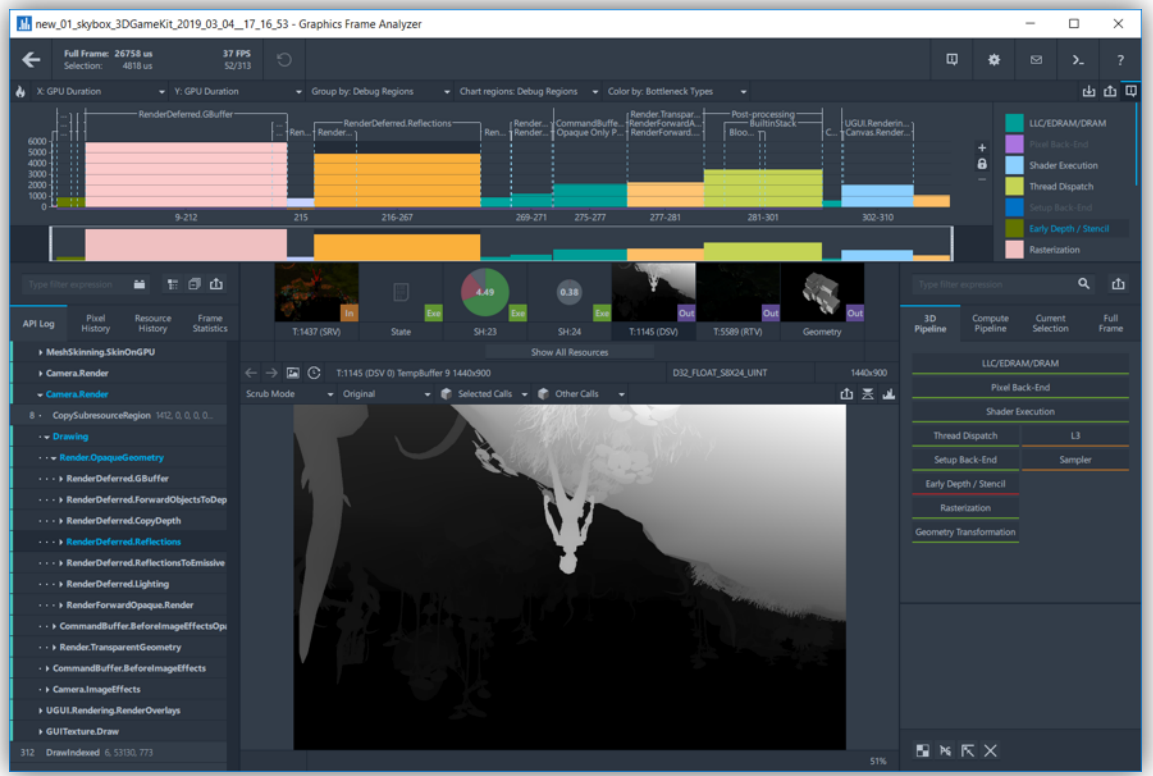
Frame Budget
(30 FPS)



Case Study: Unity 3D Game Kit

Keep peeling the onion!

Frame Budget
(30 FPS)

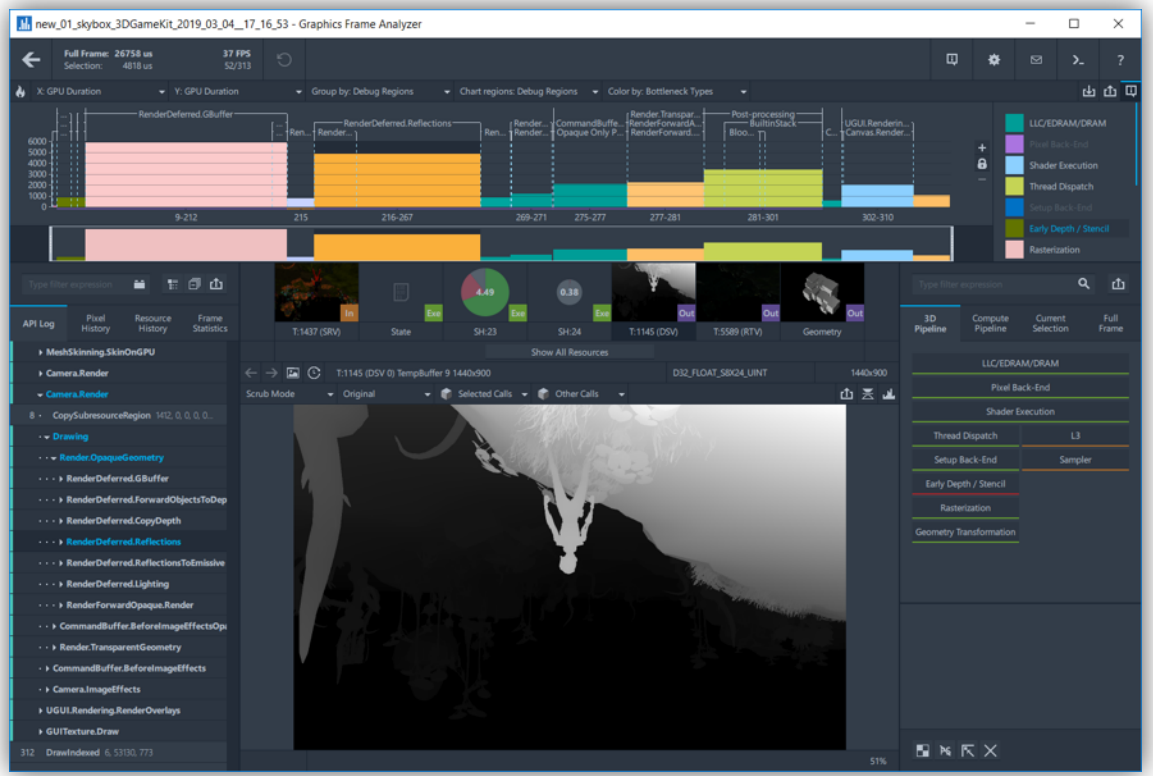


Case Study: Unity 3D Game Kit

Reflections take 4.8ms

14.5% of frame budget

Frame Budget
(30 FPS)

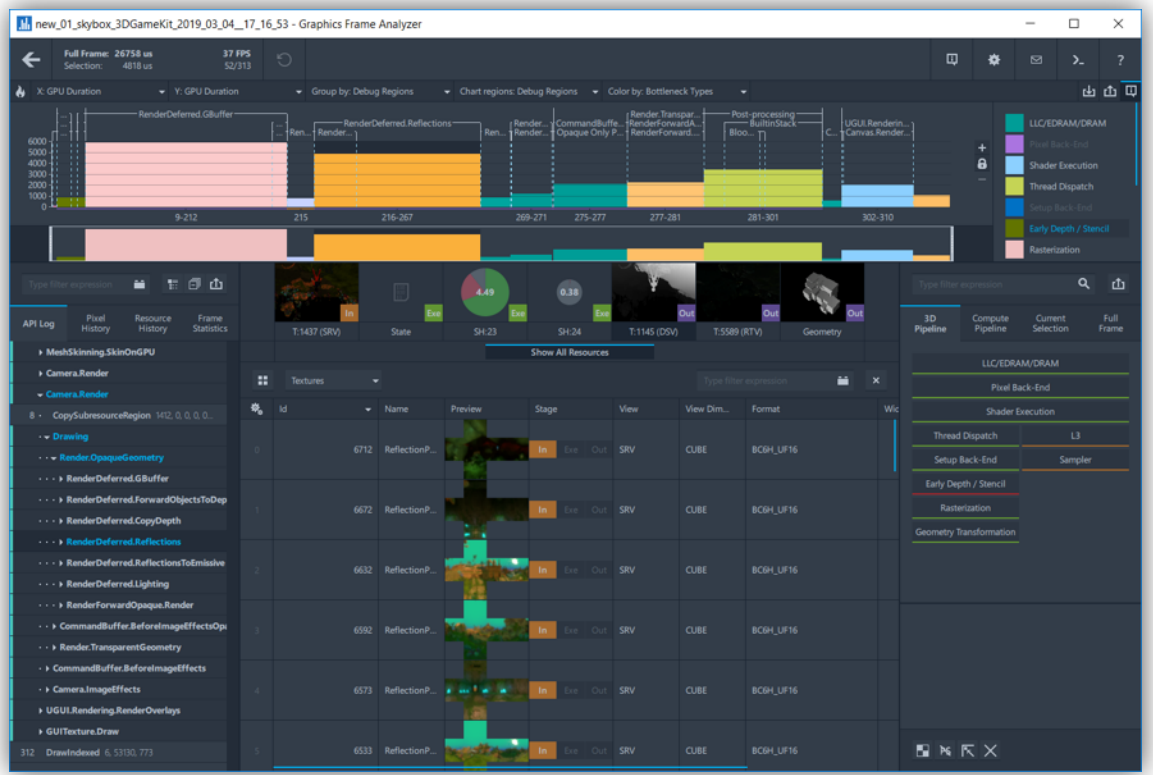


Case Study: Unity 3D Game Kit

32 Reflection probes? Can we get by with less?

14.5% of frame budget

Frame Budget
(30 FPS)

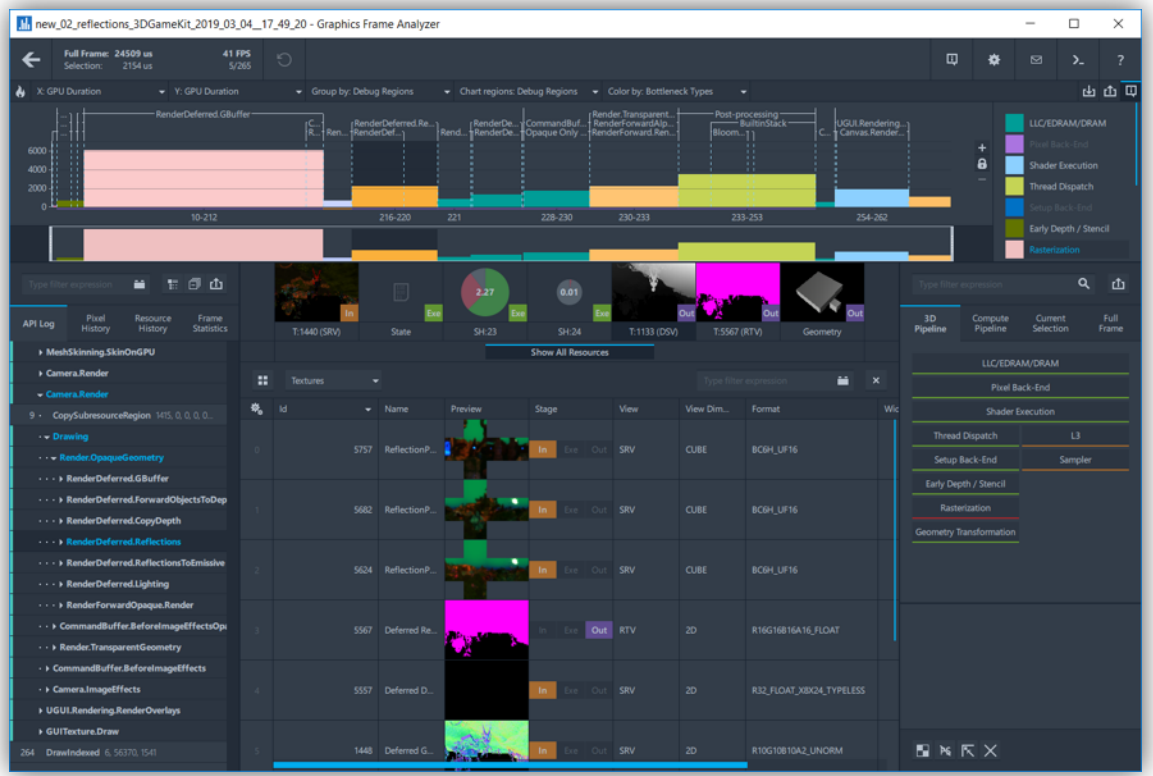


Case Study: Unity 3D Game Kit

32 to 3 reflection probes 2.1ms

6.4% of frame budget down from 14.5%

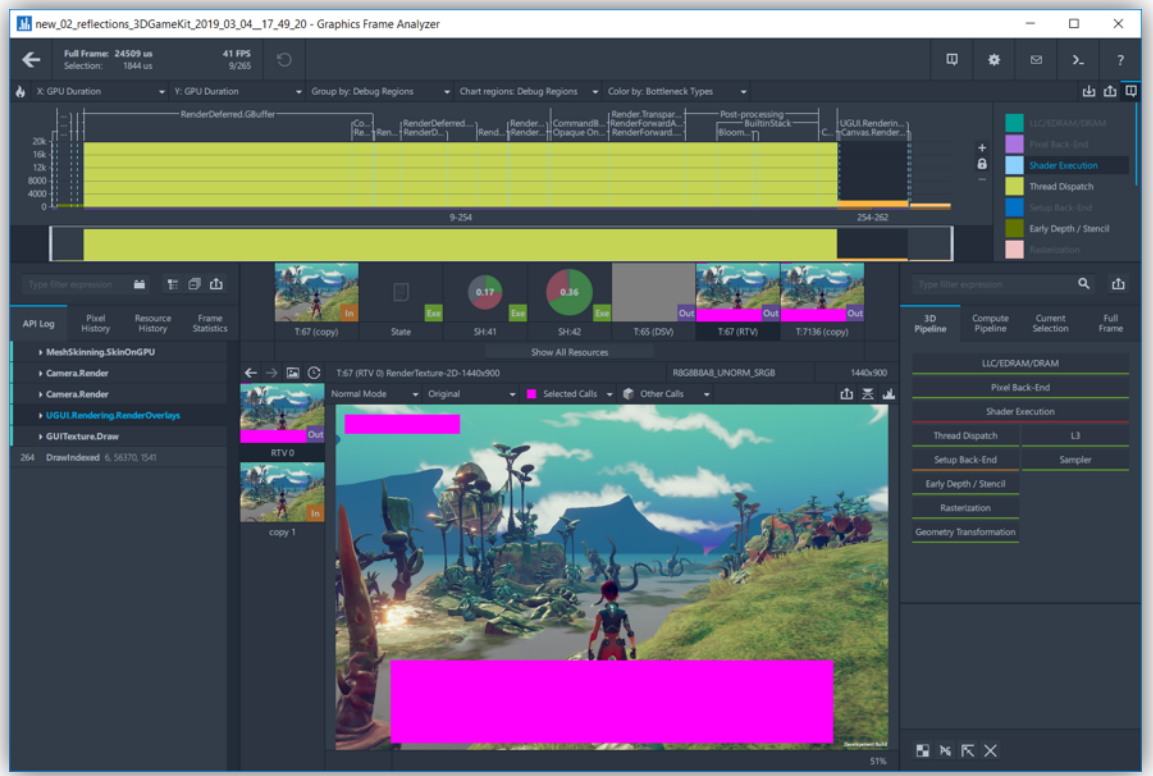
Frame Budget
(30 FPS)



Case Study: Unity 3D Game Kit

Keep peeling the onion!

Frame Budget
(30 FPS)

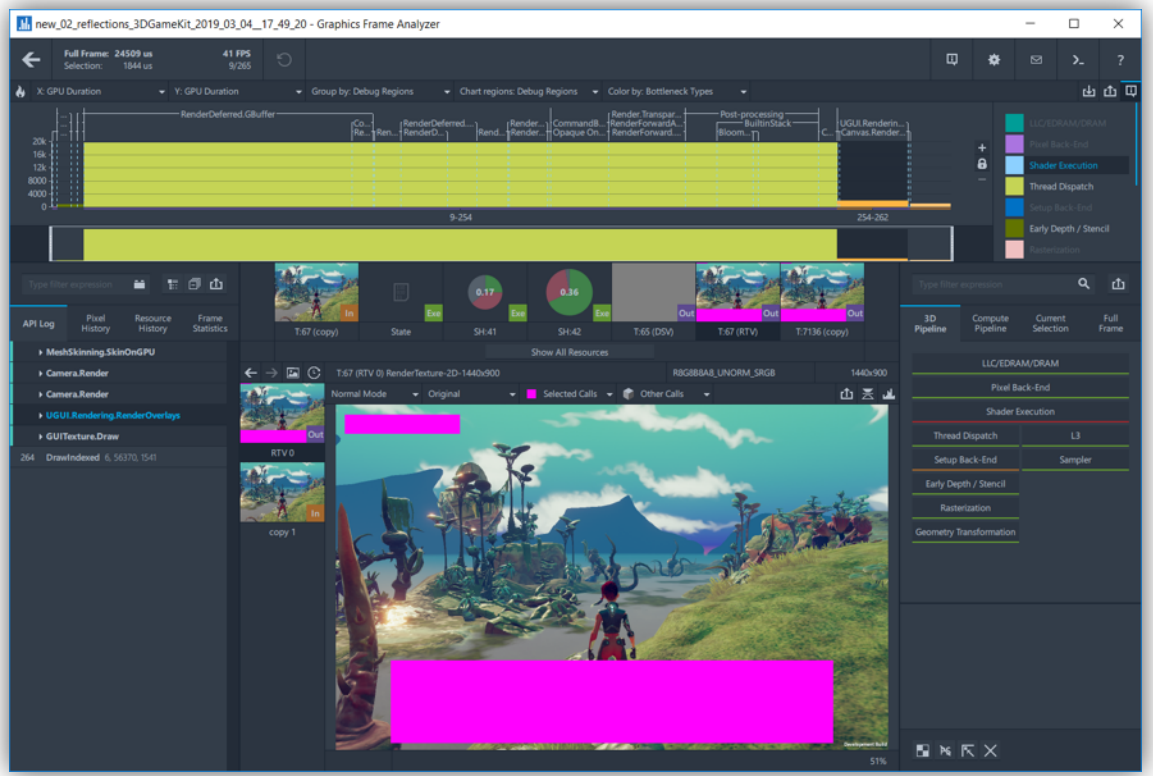


Case Study: Unity 3D Game Kit

UI rendering takes 1.8ms

8% of frame budget

Frame Budget
(30 FPS)

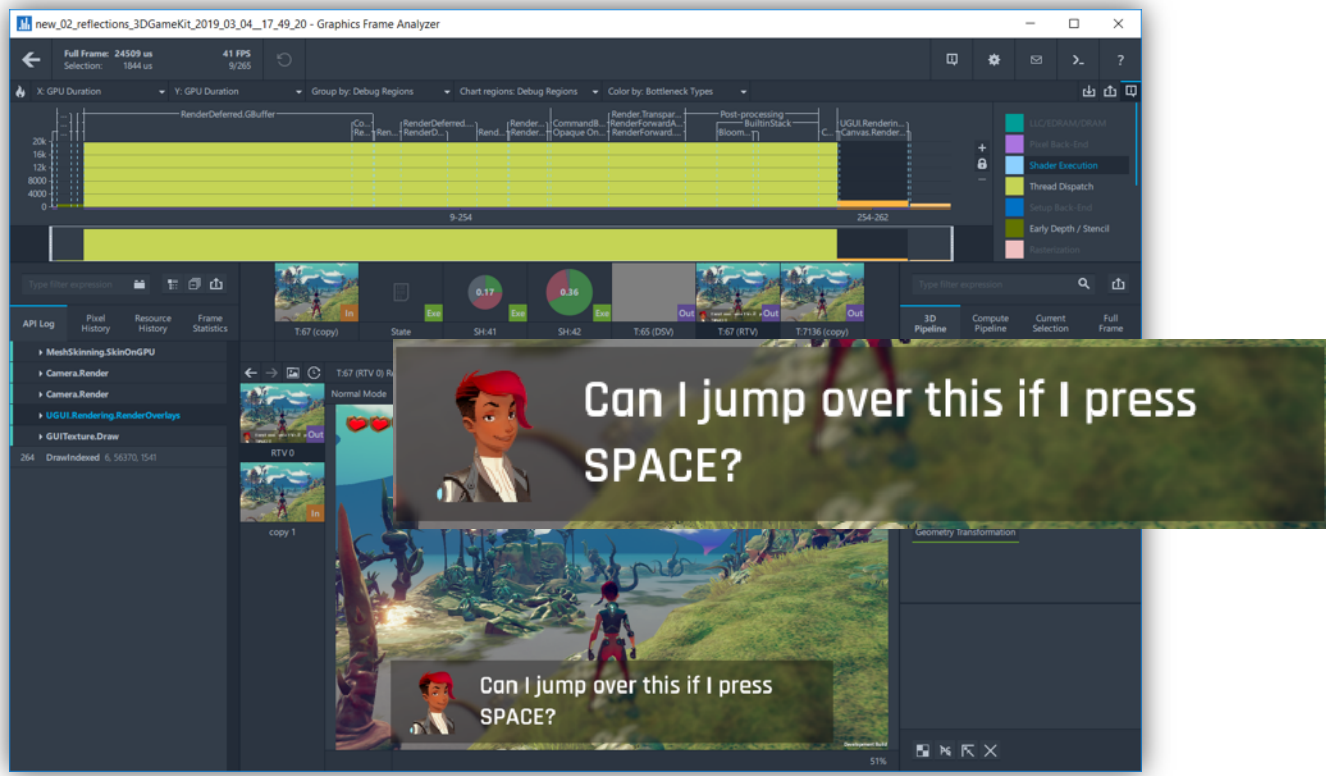


Case Study: Unity 3D Game Kit

Blur text is expensive, by how much?

8% of frame budget

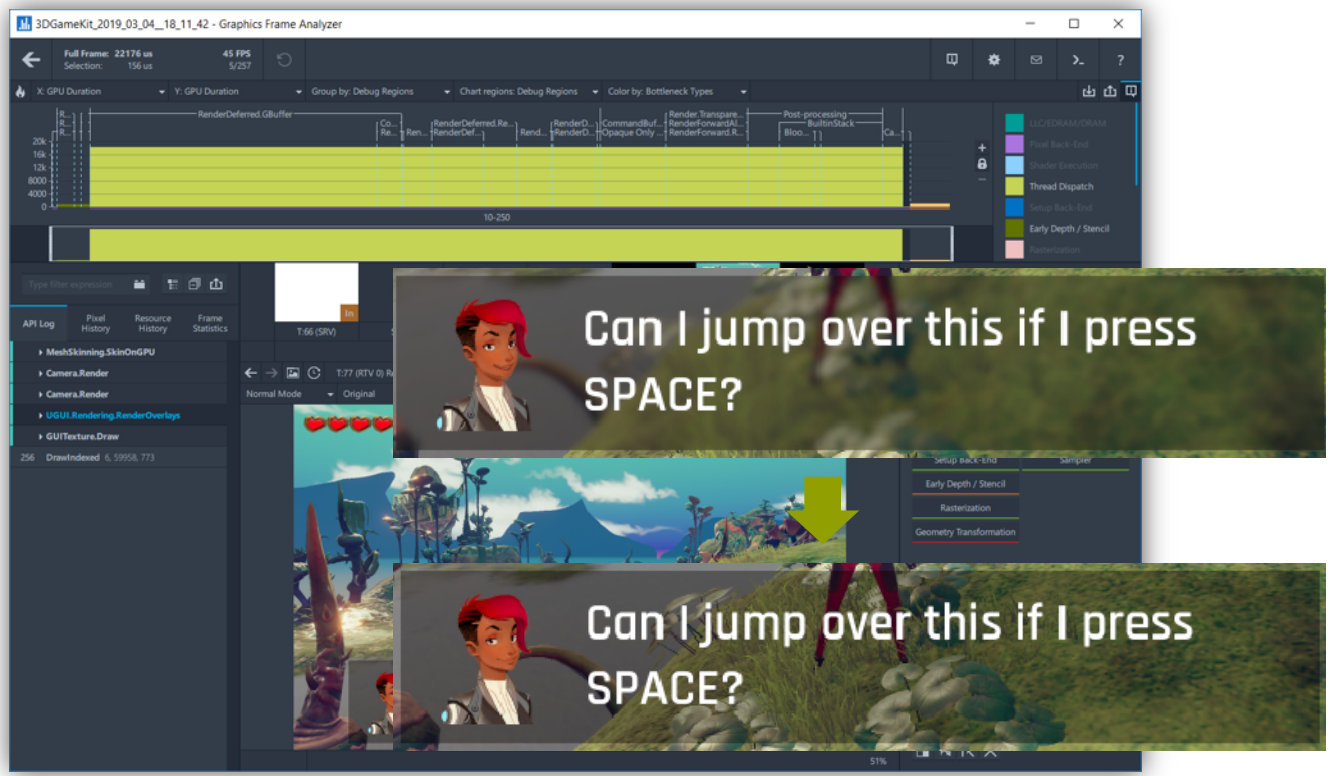
Frame Budget
(30 FPS)



Case Study: Unity 3D Game Kit

UI Rendering without blur 0.2ms 0.6% of frame budget down from 8%

Frame Budget
(30 FPS)



Case Study: Unity 3D Game Kit

Level check: Running at 38 FPS now!

Frame Budget (30 FPS)

With all the optimizations we're now running at 38 FPS

Time to start adding things back!

Lets start big: Culling distance

Case Study: Unity 3D Game Kit

Restore cull distance

Frame Budget
(30 FPS)

Before 38FPS

After 33FPS

Still some room left

Lets add it all!

Case Study: Unity 3D Game Kit

Lets add it all!

Frame Budget
(30 FPS)

Before 33FPS

After 28FPS

8% above budget but we got water reflections and shadows!

What else can we cheaply do to stay within budget?

Case Study: Unity 3D Game Kit

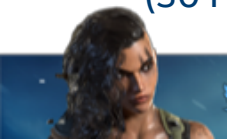
Downscale deferred render target

Frame Budget
(30 FPS)

Downscale deferred render target to a fraction of it's size

Upscale in final pass and render UI at full resolution

Back within Frame Budget!



Summary

- Scale graphics performance effectively
- Performance awareness is everyone's responsibility
- Grab Intel® GPA for free @ https://software.intel.com/en-us/gpa
- Want to learn more? Visit us at the Intel® booth
- Try it yourself!
- Twitter: @carlosadc


QUESTIONS?

Legal Notices and Disclaimers

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

The products and services described may contain defects or errors known as errata which may cause deviations from published specifications. Current characterized errata are available on request.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. No computer system can be absolutely secure. Check with your system manufacturer or retailer or learn more at [intel.com].

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit www.intel.com/benchmarks.

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

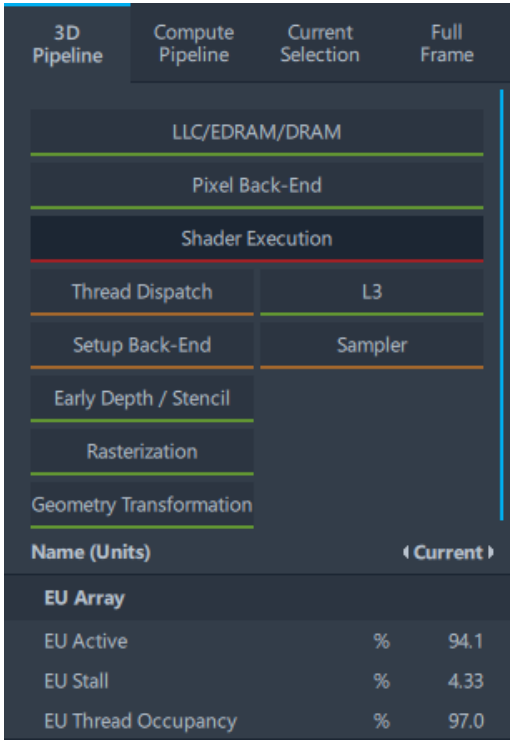
Results have been estimated or simulated using internal Intel analysis or architecture simulation or modeling, and provided to you for informational purposes. Any differences in your system hardware, software or configuration may affect your actual performance.

Intel, Core and the Intel logo are trademarks of Intel Corporation in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others

Shader Execution

- Bottleneck in Shader Execution
- GPU Execution Units (EU's) very active
- Reduce shader complexity
- Reduce GPRs used in shader to avoid register spilling



Shader Execution

- Simple Fragment Shader Experiment
- Quick check for ROI on optimizing shader
- Reduce shader optimization iteration time with GPA's Live Shader Analysis
  - Modify shader
  - Replay scene
  - Recalculate metrics