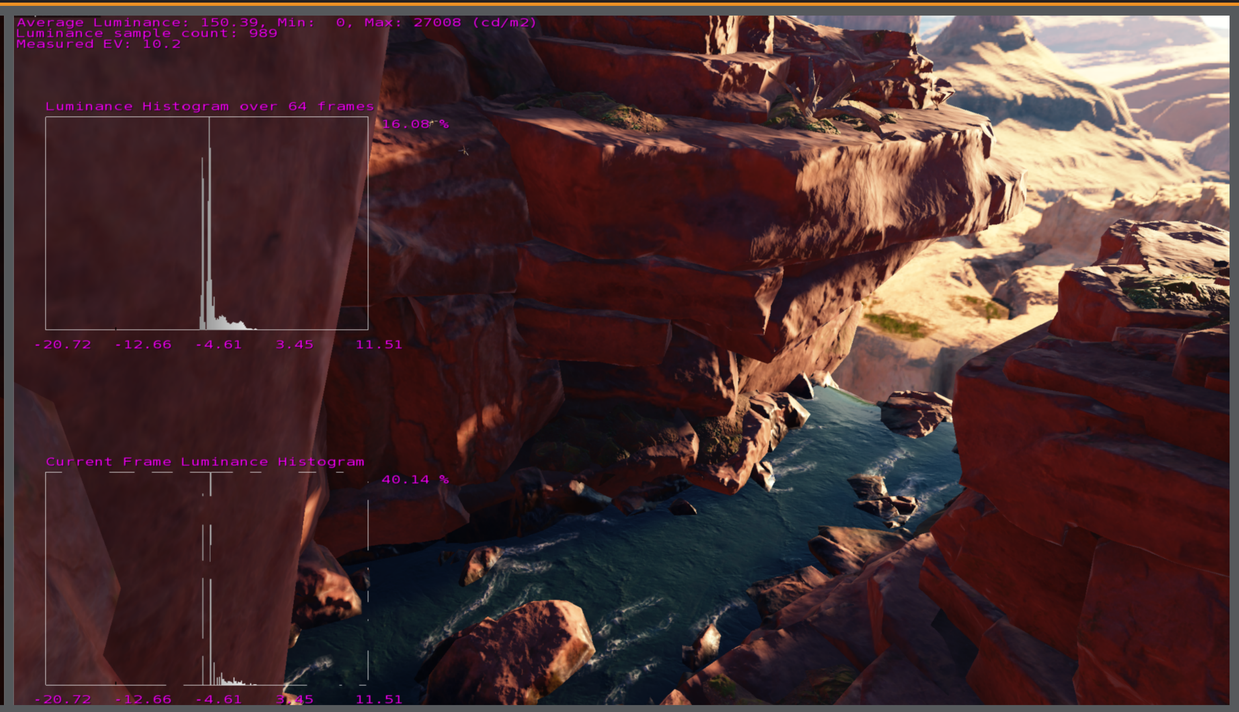
Fast Eye-Adaptation for Mobile Applications

Measuring average global/local luminance of the final picture on tiled architecture GPUs

Morteza Mostajab, Theodor Mader

Crytek GmbH



Objective

To obtain an efficient solution for measuring the global/local average luminance value of an HDR image, that can be used to enhance its quality by adjusting its overall brightness. Vulkan's subpasses can exploit the performance of this method, and reduce the cost imposed on the GPU.

Problem Definition

- VR gaming and app development has expanded over the past few years. One of the recent efforts for making user experience smoother is release of standalone mobile head-mounted displays (HMDs) in the market. These headsets benefit from a mobile system on the chip (SoC) that is used for producing and rendering virtual reality content.
- The proposed solution is used in Crytek's game : The Climb released on Oculus Quest. Oculus Quest is a mobile VR HMD that has a Qualcomm Snapdragon 835 system on the chip (SoC), with one display per eye of 1440x1600 resolution and 72 Hz refresh rate.
- Most of the levels are in natural environments with natural day/night light thus the final image is having a high dynamic range; sunny midday, inside a cave with the sky visible from its opening, and night with bright lanterns flying in the sky are a few examples of these environments. The output image without a proper adaptation and tone mapping may have over/under-exposing problems.
- Latest CRYENGINE calculates the average luminance by down-sampling the HDR target to a single pixel. The luminance of the single-pixel is used to apply global eye-adaptation.

Relevant Hardware and Vulkan Specifications

- Today's mobile SoCs are benefiting from tiled-architecture with a fast, small block of tiled memory connected through a bus to system memory. The bandwidth between these two memories is slower compared to today's hardware used in PC and consoles. As the down-sampling of the HDR target is done in multiple passes, it involves the transfer of a significant amount of data back-and-forth between tiled and system memories. This makes the common luminance measurement a performance bottleneck for the rendering pipeline. Also, it forces the rendering pipeline to store the data in an intermediate roughly accurate representation (R10G10B10A2 or R16G16B16A16).
- Vulkan API allows the programmer to have low-level control over hardware utilization. Vulkan's RenderPass with multiple Subpasses can be used to get the most out of tiled-architecture by applying multiple subpasses on the data available in fast tiled memory, without resolving it to system memory in an intermediate stage. It also gives the programmer the ability to define the behavior of the hardware when it comes to loading/storing data from/in system memory. This feature tremendously reduces bandwidth usage .

Contact Information

- Email: seyedmorteza@crytek.com

Rendering Pipeline

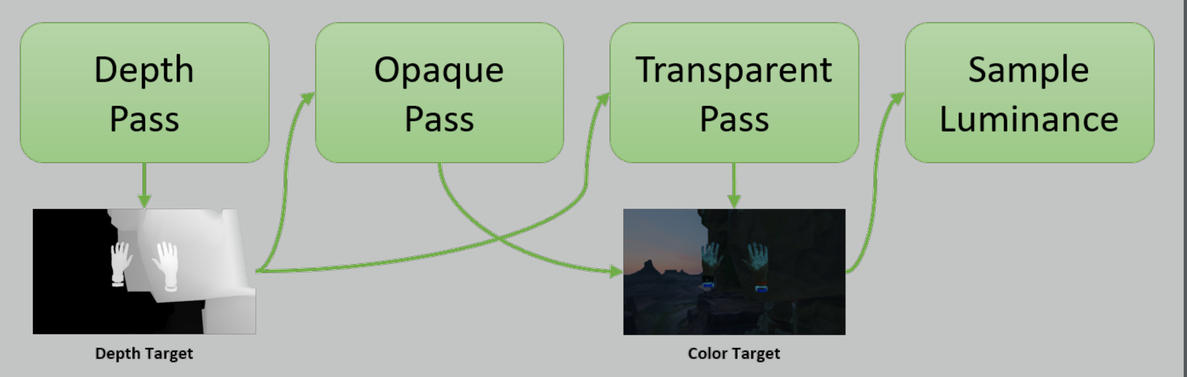


Figure 1:Rendering pipeline stages

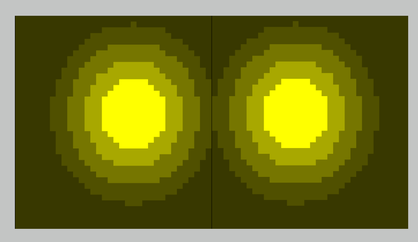


Figure 2:Fixed Foveated Density Map

- The scene is rendered using a Forward approach with a depth pre-pass (Figure 1). Objects are sorted based on their pipeline state object (PSO), and drawn front-to-back.
- Fixed foveated rendering approach is used. The fragment shading rate is visualized by varying intensity of the yellow color in Figure 2.
- Inline Tonemapping: In our game, tone mapping was done inside opaque and transparent passes after shading, instead of a separate pass. This helped to eliminate an intermediate floating-point representation of HDR target, which in turn demanded high memory bandwidth. Tone mapping consists of:
  - **Auto-Exposure**: Automatically adjusting the overall picture's brightness.
  - **Filmic Tone-mapping**: Maps HDR to LDR values using Hable's curve.
  - **Color grading**: Re-maps the final color based on a color chart.

Solution: Stochastic Luminance Sampling and Adaptive Encoding

- Renders a specific amount of uniformly distributed points on the screen, as well as output luminance values to a host-accessible buffer. The distribution of points changes every frame.
- Luminance values are calculated, compressed, and stored in color target's alpha-channel in Opaque/Transparent passes. In Sample luminance step, they are decompressed and written to the host-accessible buffer.
- To avoid any interruption in the device's work for reading back the luminances on host, this operation is latent for a few frames.
- Host collects the read-back data for a specific number of frames, and calculates the global/local average luminance values.
- Statistics of levels of brightness are represented as a histogram of the logarithm of luminance values over a specific number of frames.
- Histogram equalization is used as an alternative value to adapt image overall brightness, instead of average luminance.
- Luminance sampling takes 0.1-0.2 ms on GPU when it runs as a separate render pass, and less than 0.1ms when it runs as a subpass.

References

[1] Gregory Ward Larson, Holly Rushmeier, and Christine Piatko.
A visibility matching tone reproduction operator for high dynamic range scenes.
IEEE Transactions on Visualization and Computer Graphics, 3(4):291–306, October 1997.