Fast Eye-Adaptation for High Performance Mobile Applications

Morteza Mostajab*
seyedmorteza@crytek.com
Crytek GmbH

Theodor Mader*
theodor@crytek.com
Crytek GmbH

Figure 1: Two screenshots from The Climb, a Crytek game, on Oculus Quest. Tone mapping plays an important role in enhancing the final image quality of natural environments with sun and moonlight: On the left is a midday rock-climbing environment, and On the right, the player can be seen inside a cave lit by the setting sun. Without proper tone mapping, both views would contain under/over exposed areas. One of the steps involved in tone mapping is to adapt the colors of the final image to correct the exposure of these areas; this paper proposes a high performance approach for eye-adaptation on mobile head-mounted displays, making it usable on mobile VR devices with their special performance requirements.

ABSTRACT

The virtual reality (VR) industry has been growing over the last decade. Commercial head-mounted displays (HMD) are released in the market, different kinds of applications and games are released for them. Although the VR experience has seen a lot of improvement, one of the limiting factors is the lack of VR headset mobility. Mobile VR HMD is one of the efforts to address this problem. While this improves user experience significantly, the rendering pipeline of games needs to be revisited and adapted to the limitations and benefits of the mobile architecture.

Tonemapping is one of the final stages in most rendering pipelines, and its main purpose is to enhance the quality of the final image perceived by the user. Measuring average luminance is a critical step of tone mapping; however, considering VR frame-rate requirements, the traditional average luminance measurement method used in today's PCs/consoles is not fast enough on mobile chips. This paper presents an efficient solution for approximating the average luminance value of an HDR image, by constantly sampling it on the GPU and processing them on the CPU. This approach outperforms traditional methods, as it doesn't rely on down-sampling the HDR target. Also, Vulkan's subpasses can exploit the performance of this method, and reducing the cost imposed on the GPU.

Woodstock '18, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

CCS CONCEPTS

• **Applied computing** → *Media arts.*

KEYWORDS

mobile rendering, post-processing, tone mapping, eye adaptation

ACM Reference Format:
Morteza Mostajab and Theodor Mader. 2018. Fast Eye-Adaptation for High Performance Mobile Applications. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

VR gaming and app development has expanded over the past few years. Previously, interwoven cables connecting head-mounted displays (HMDs) to the PC prevented user experience from being smooth; recent efforts to address this problem include standalone mobile HMDs benefiting from a mobile system on the chip (SoC). One such example is Oculus Quest, which has a Qualcomm Snapdragon 835 SoC with a display of 1440x1600 resolution and 72 Hz refresh rate per eye. The solution proposed in this paper is used in Crytek's The Climb for Oculus Quest. Most of its levels are in natural environments with natural day/night light causing the image to have 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 proper adaptation and tone mapping may

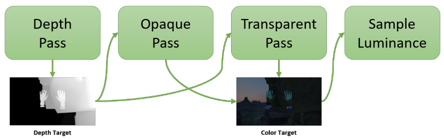

Figure 2: The Climb's Rendering Pipeline Stages

show over/under exposure. Traditional Cryengine tone mapping enhances the image quality, by adapting its colors to an average luminance of the final image. Average luminance is calculated by down-sampling the HDR target to a single pixel; the luminance of the single pixel is used to apply global adaption of the image's color, known as eye-adaptation.

Today's mobile SoCs are benefiting from a tiled-architecture with a fast, small block of tiled memory connected through a bus to the system memory. The bandwidth between these two memories is slower in comparison to today's hardware used in PC and consoles. As the down-sampling of the HDR target is done in multiple passes, it involves transferring a significant amount of data back-and-forth between these two types of memories. Also, it forces the rendering pipeline to store the data in an intermediate roughly accurate representation (e.g. R10G10B10A2 or R16G16B16A16) that when transferred 72 times per frame for each eye, consumes significant bandwidth. The Vulkan graphics API allows low-level control over utilizing tiled memory, and over its transactions with system memory. In Vulkan, a renderPass can be defined with multiple Subpasses that operate on the tiled memory without the need to resolve the data to system memory.

The main contribution of this paper is to provide 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.

2 RENDERING PIPELINE

Since our product is running on a mobile SoC and must fulfill today's VR headset requirements of a high resolution and refresh rate, we've developed a simple but high-performance pipeline. The proposed pipeline is a forward rendering pipeline with a depth pre-pass stage executing in advance. Draw calls are sorted based on their PSO, and drawn front-to-back. The following passes constitute the high-performance graphic pipeline (Figure 2):

- **Depth-Pass**: Renders all opaque object draw calls, and writes out their depth in the depth target.
- **Opaque-Pass**: Shades visible fragments produced from drawing of opaque objects and outputs the results into the color target.
- **Transparent-Pass**: Shades and blends transparent objects with the color target.
- **Sample Luminance**: Samples luminance of the final target, and write it to a host-accessible buffer.

To improve the performance, a variable rate shading fixed foveation rendering approach is used. Figure 3 visualizes the fragment shading rate by the varying intensity of the color yellow.

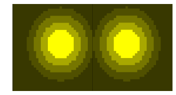

Figure 3: Left and right eye shading rate is visualized by varying the intensity of the color yellow for left and right eyes.

In The Climb, tone mapping is done directly inside the Opaque and Transparent passes on the shading output. This is done to eliminate the need to store an intermediate floating-point HDR target in the system memory, which saves a significant amount of memory bandwidth. Our tone mapping approach consists of three main stages:

- **Auto-Exposure** that automatically adjusts the overall brightness of the final image.
- **Filmic Tone-mapping** which maps the HDR values of the final image to LDR values. The Climb uses the well known Hable curve [1].
- **Color grading** that re-maps the final color, based on a color chart that is pre-processed by an artist.

3 SOLUTION: STOCHASTIC LUMINANCE SAMPLING AND ADAPTIVE ENCODING

In our proposed solution, luminance values are sampled randomly from the final image and transferred to the CPU. The average luminance is calculated by the CPU, based on collected samples over a specific number of frames. Luminance values are sampled by rendering several uniformly distributed point primitives on the screen with a fragment shader which outputs the calculated luminance to a host accessible buffer. In The Climb, luminance values are calculated and stored in the alpha channel of the color target in Opaque/Transparent pass. The distribution of sample points changes every frame, in order to cover more areas of the image over multiple frames. Since we use an RGBA8 color target, luminance values must remapped and compressed to minimize loss of precision. To avoid any interruption for the GPU's work, the luminance values are read back with a latency of few frames.

Collection of read-back data over a few frames (e.g. 72 frames) is used to calculate the average luminance value for the auto-exposure stage of tone mapping. Furthermore, a histogram of levels of brightness is established for the collected data. The histogram equalization [2] method is utilized to calculate an alternative value to adapt the overall picture's brightness, instead of average luminance. This value improves the final visual quality, since it is less sensitive to outliers (for example, when a small region of very bright pixels is visible on screen, such as when the sun is visible during sunset).

The proposed approach is friendly to mobile SoC tiled-architectures. When it runs as a separate renderpass, it has a constant cost of 0.1-0.2ms on Snapdragon 635. Its cost can be reduced to less than 0.1 ms by sampling in a subpass from tiled memory.

REFERENCES

[1] John Hable. 2010. Uncharted 2: HDR Lighting. (2010).

[2] Gregory Ward Larson, Holly Rushmeier, and Christine Piatko. 1997. A Visibility Matching Tone Reproduction Operator for High Dynamic Range Scenes. *IEEE Transactions on Visualization and Computer Graphics* 3, 4 (Oct. 1997), 291–306. https://doi.org/10.1109/2945.646233