Energy-Efficient Video Processing for Virtual Reality

Yue Leng and Jian Huang
University of Illinois at Urbana–Champaign

Chi-Chun Chen, Qiuyue Sun, and Yuhao Zhu
University of Rochester

***Abstract*—Virtual reality (VR) has huge potential to enable radically new applications, behind which spherical panoramic video processing is one of the backbone techniques. However, current VR systems reuse the techniques designed for processing conventional planar videos, resulting in significant energy inefficiencies. Our characterizations show that operations that are unique to processing 360° VR content constitute 40% of the total processing energy consumption. We present EVR, an end-to-end system for energy-efficient VR video processing. EVR recognizes that the major contributor to the VR tax is the projective transformation (PT) operations. EVR mitigates the overhead of PT through two key techniques: semantic-aware streaming on the server and hardware-accelerated rendering on the client device. Real system measurements show that EVR reduces the energy of VR rendering by up to 58%, which translates to up to 42% energy saving for VR devices.**

Virtual reality (VR) has profound social impact in transformative ways. For instance, immersive VR experience is shown to reduce patient pain more effectively than traditional medical treatments, and is seen as a promising solution to the opioid epidemic. One of the key use-cases of VR is 360° video processing. Unlike conventional planar videos, 360° videos embed panoramic views of the scene. As users change the viewing angle, the VR device renders different parts of the scene, mostly on a head-mounted display (HMD), providing an immersive experience.

A major challenge in VR video processing today is the excessive power consumption of VR devices. Our measurements show that rendering 720p VR videos in 30 frames per second (FPS) consistently consumes about 5 W of power, which is twice as much power than rendering

Digital Object Identifier 10.1109/MM.2020.2985692
Date of publication 6 April 2020; date of current version 22 May 2020.

Published by the IEEE Computer Society

conventional planar videos and exceeds the thermal design point (TDP) of typical mobile devices.[4] The device power requirement will only grow as users demand higher frame-rate and resolution, presenting a practical challenge to the energy- and thermal-constrained mobile VR devices.

The excessive device power is mainly attributed to the fundamental mismatch between today's VR system design philosophy and the nature of VR videos. Today's VR video systems are designed to reuse well-established techniques designed for conventional planar videos.[1] This strategy accelerates the deployment of VR videos, but causes significant energy overhead. More specifically, VR videos are streamed and processed as conventional planar videos. As a result, once on-device, each VR frame goes through a sequence of spherical–planar projective transformations (PT) that correctly render a user's current viewing area on the display. The PT operations are pure overhead uniquely associated with processing VR videos—operations that we dub "VR tax." Our characterizations show that "VR tax" is responsible for about 40% of the processing energy consumption, a lucrative target for optimizations.

A major challenge in VR video processing today is the excessive power consumption of VR devices. Our measurements show that rendering 720p VR videos in 30 frames per second (FPS) consistently consumes about 5 W of power, which is twice as much power than rendering conventional planar videos and exceeds the thermal design point (TDP) of typical mobile devices.

We present EVR, an end-to-end system for energy-efficient VR video processing. EVR recognizes that the major contributor to the VR tax is the PT operations. EVR mitigates the overhead of PT through two key techniques: semantic-aware streaming (SAS) on the server and hardware-accelerated rendering (HAR) on the client device. EVR uses SAS to reduce the chances of executing PT on VR devices by prerendering 360° frames in the cloud. Different from conventional prerendering techniques, SAS exploits the key semantic information inherent in VR content that is previously ignored. Complementary to SAS, HAR mitigates the energy overhead of on-device rendering through a new hardware accelerator that is specialized for PT. We implement an EVR prototype on an Amazon AWS server instance and an NVIDA Jetson TX2 board combined with a Xilinx Zynq-7000 FPGA. Real system measurements show that EVR reduces the energy of VR rendering by up to 58%, which translates to up to 42% energy saving for VR devices.

ENERGY CHARACTERIZATIONS

A VR system involves two distinct stages: capture and rendering. VR videos are captured by special cameras, which generate 360° images that are best presented in the spherical format. The spherical images are then projected to planar frames through one of the spherical-to-planar projections, such as the equirectangular projection. The planar video is either directly live-streamed to client devices for rendering (e.g., broadcasting a sports event), or published to a content provider, such as YouTube or Facebook, and then streamed to client devices upon requests. Alternatively, the streamed videos can also be persisted in the local storage on a client device for future playback. This article focuses on client-side VR content rendering, i.e., after a VR video is captured, because rendering directly impacts VR devices' energy efficiency.

Rendering VR videos consumes excessive power on the VR device, which is particularly problematic as VR devices are energy and thermal constrained. This section characterizes the energy consumption of VR devices. Although there are many prior studies that focused on energy measurement of mobile devices such as smartphones and smartwatches, this is the first such study that specifically focuses on VR devices. We show that the energy profiles between VR devices and traditional mobile devices are different.

We conduct studies on a recently published VR video dataset, which consists of head movement traces from 59 real users viewing different 360° VR videos on YouTube.[3] We replay the traces

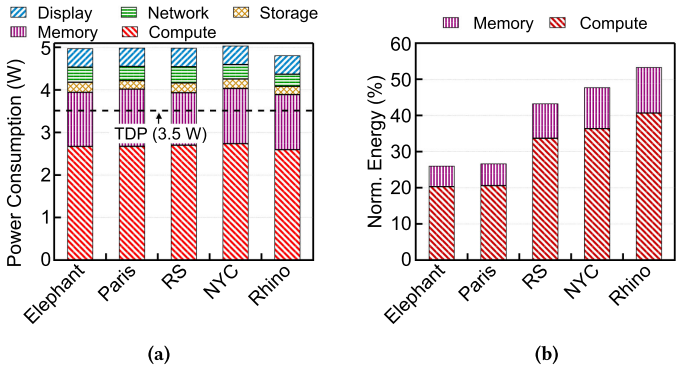

Figure 1. Power and energy characterizations of VR device. (a) Power distribution across the major components in a VR device. (b) Contribution of the PT operations to the compute and memory energy.

to mimic realistic VR viewing behaviors. We assemble a custom VR device based on the NVIDIA Jetson TX2 development board in order to conduct fine-grained power measurements on the hardware, which are infeasible in off-the-shelf VR devices. We refer readers to the "Evaluation Methodology" section for a complete experimental setup.

Power and Energy Distribution

We breakdown the device power consumption into five major components: display, network (WiFi), storage (eMMC), memory (DRAM), and compute (SoC). The storage system is involved mainly for temporary caching. We show the power distribution across the five components for the five VR video workloads in Figure 1(a). The power consumption is averaged across the entire viewing period. Thus, the power consumption of each component is proportional to its energy consumption.

We make two important observations. First, the device consistently draws a power consumption of about 5 W across all five VR videos. As a comparison, the TDP of a mobile device, i.e., the power that the cooling system is designed to sustainably dissipate, is around 3.5 W,[4] clearly indicating the need to reduce power consumption.

Second, unlike traditional smartphone and smartwatch applications where network, display, and storage consume significant energy,[2,7,9] the energy consumptions of the three components in a VR device are relatively insignificant, contributing to only about 9%, 7%, and 4% of the total energy consumption, respectively. This indicates that optimizing network, display, and storage would lead to marginal energy reductions. More lucrative energy reductions come from optimizing compute and memory.

Contribution of VR Operations We further find that energy consumed by executing operations that are uniquely associated with processing VR videos constitutes a significant portion of the compute and memory energy. Such operations mainly consist of PT operations. We show the energy contribution of the PT operations to the total compute and memory energy as a stacked bar chart in Figure 1(b). On average, PT contributes to about 40% of the total compute and memory energy, and is up to 53% in the case of video Rhino. The PT operations exercise the SoC more than the DRAM as is evident in their higher contributions to compute energy than memory energy.

Overall, our results show that the PTs would be an ideal candidate for energy optimizations.

ENERGY-EFFICIENT VR WITH EVR

The goal of EVR is to reduce energy consumption of VR devices by optimizing the core components that contribute to the "VR tax." We present an end-to-end energy-efficient VR system with optimization techniques distributed across the cloud server and the VR client.

Semantics-Aware Streaming

Our key idea is to leverage video-inherent semantic information that is largely ignored by today's VR servers. SAS specifically focuses on one particular form of semantic information: visual object. We show that users tend to focus on objects in VR content, and object trajectories provide a proxy for predicting user viewing areas. We leverage a recently published VR video data set,[3] which consists of head movement traces from 59 real users viewing different 360° VR videos on YouTube. We further confirm that users track the same set of objects across frame rather than frequently switching objects. Specifically, we measure the time durations during which users keep

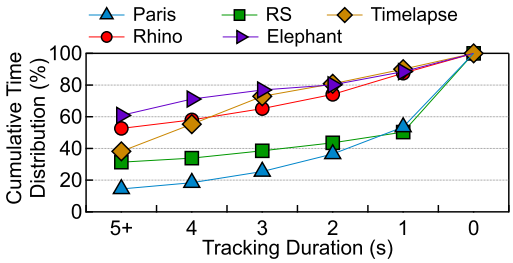

Figure 2. Cumulative distribution of tracking durations.

tracking the movement of the same object, and show the results in Figure 2 as a cumulative distribution plot. On average, users spend about 47% of time tracking an object for at least 5 s.

The near 100% frame coverage in many videos as the number of identified objects increases indicates that the server can effectively predict user viewing area solely based on the visual objects without sophisticated client-side mechanisms such as using machine learning models to predict users' head movement.[5,10] This observation frees the resource-constrained VR from performing additional work and simplifies the client design.

SAS has two major components: First, a static and offline analysis component that extracts objects from the VR video upon ingestion and generates a set of FOV videos that could be directly visualized once on a VR device; second, a dynamic and runtime serving component that streams FOV videos on demand to the VR device. We augment the new FOV video with metadata that corresponds to the head orientation for each frame. Once the FOV video together with its associated metadata is on the client side and before a FOV frame is sent to the display, the VR client compares the desired viewing area indicated by the head motion sensor with the metadata associated with the frame. If the two match, the client directly visualizes the frame on the display, bypassing the PT operations. Otherwise, the client system requests the original video segment from the cloud, essentially falling back to the normal VR rendering mode.

Note that in our current design, we make the simplification that the client will always fall back to the regular VR processing flow upon FOV-miss and restart streaming FOV videos based on user's current focus. One could also imagine other design alternatives. For instance, the client could send the current (desired) FOV to the cloud service, which returns another FOV video if there happens to be one that matches the desired FOV. We leave it as future work to explore the full design space of the dynamic component.

Hardware-Accelerated Rendering

We propose a new hardware accelerator, PT engine (PTE), that performs efficient PTs. We design the PTE as an SoC IP block that replaces the GPU and collaborates with other IPs such as the Video Codec and Display Processor for VR video rendering. Figure 3 shows how PTE fits into a complete VR hardware architecture. The PTE takes in frames that are decoded from the video codec, and produces FOV frames to the frame buffer for display. If a frame is already prepared by the cloud server as a projected FOV frame, the PTE sends it to the frame buffer directly; otherwise the input frame goes through the PTE's datapath to generate the FOV frame. The GPU can remain idle during VR video playback to save power.

The bulk of the PTE is a set of PT units (PTU) that exploits the pixel-level parallelism. The pixel memory (P-MEM) holds the pixel data for the incoming input frame, and the sample memory (S-MEM) holds the pixel data for the FOV frame that is to be sent to the frame buffer. The PTE uses DMA to transfer the input and FOV frame data. The PTE also provides a set of

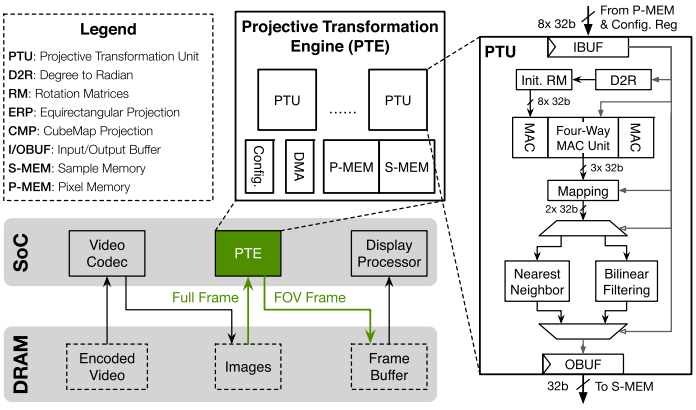

Figure 3. Overview of the augmented hardware architecture.

memory-mapped registers for configuration purposes. The configurability allows the PTE adapt to different popular projection methods and VR device parameters, such as FOV size and display resolution. The configurability ensures PTE's flexibility without the overhead of general-purpose programmability that GPUs introduce.

The P-MEM and S-MEM must be properly sized to minimize the DRAM traffic. Holding the entire input frame and FOV frame would require the P-MEM and S-MEM to match the video resolution (e.g., 4K) and display resolution (e.g., 1440p) respectively, requiring tens of MBs on-chip memories that are prohibitively large in practice. Interestingly, we find that the filtering step (the only step in the PT algorithm that access pixel data) operates much like a stencil operation that possesses two properties. First, PT accesses only a block of adjacent pixels for each input. Second, the accessed pixel blocks tend to overlap between adjacent inputs. Thus, the P-MEM and S-MEM are designed to hold several lines of pixels in the input and FOV frame, which is similar to the line buffer used in image signal processor (ISP) designs.[6]

EVR Implementation

Building on top of the two optimizing primitives, SAS and HAR, we design EVR. EVR includes a cloud component and a client component. The cloud component extracts object semantics from VR videos upon ingestion, and prerenders a set of miniature videos that contain only the user viewing areas and that could be directly rendered as planar videos by leveraging the powerful computing resources on the cloud. The client component retrieves the miniature video with object semantics, and leverages the specialized accelerator for energy-efficient on-device rendering if the original full video is required. For VR applications whose content comes from panoramic videos available on the VR devices, the HAR can accelerate the video rendering with lower energy overhead. We implement EVR in a prototype system, where the cloud service is hosted on an AWS instance while the client is deployed on a customize platform that combines the NVIDIA TX2 and Xilinx Zynq-7000 development boards, which can represent a typical VR client device.

EVALUATION

Evaluation Methodology

Usage Scenarios. We evaluate three EVR variants, each applies to a different use case, to demonstrate EVR's effectiveness and general applicability. The three variants are as follows.

- S: leverages SAS without HAR.
- H: uses HAR without SAS.
- S+H: combines the two techniques.

Energy Evaluation Framework Our energy evaluation framework considers the five important components of a VR device: network, display, storage, memory, and compute. The network, memory, and compute power can be directly measured from the TX2 board through the onboard Texas Instruments INA 3221 voltage monitor IC. We also use a 2560 × 1440 AMOLED display that is used in Samsung Gear VR and its power is measured in our evaluation. We estimate the storage energy using an empirical eMMC energy model[7] driven by the storage traffic traces.

Baseline We compare against a baseline that is implemented on the TX2 board and that does not use SAS and HAR. The baseline is able to deliver a real-time (30 FPS basis) user experience. Our goal is to show that EVR can effectively reduce the energy consumption with little loss of user experience.

Benchmark To faithfully represent real VR user behaviors, we use a recently published VR video data set,[3] which consists of head movement traces from 59 real users viewing different 360° VR videos on YouTube. The videos have a 4K (3840 × 2160) resolution, which is regarded as providing an immersive VR experience. The data set is collected using the Razer Open Source Virtual Reality HDK2 HMD with an FOV of 110° × 110°, and records users' real-time head movement traces. We replay the traces to emulate readings from the IMU sensor and thereby mimic realistic VR viewing behaviors. This trace-driven methodology ensures the reproducibility of our results.

Results

Energy Reductions On average, S and H achieve 22% and 38% compute energy savings, respectively. S+H combines SAS and HAR and delivers an average 41%, and up to 58%, energy saving. The compute energy savings across applications are directly proportional to the PT operation's contributions to the processing energy, as shown in Figure 1(b). For instance, Paris and Elephant have lower energy savings because their PT operations contribute less to the total compute energy consumptions.

The trend is similar for the total device energy savings. S+H achieves on average 29% and up to 42% energy reduction. The energy reduction increases the VR viewing time, and also reduces the heat dissipation and, thus, provides a better viewing experience.

User Experience Impact We also quantify user experience both quantitatively and qualitatively. Quantitatively, we evaluate the percentage of FPS degradation introduced by EVR compared to the baseline. We show that the FPS drop rate averaged across 59 users is only about 1%. Lee *et al.* reported that a 5% FPS drop is unlikely to affect user perception.[8] We assessed qualitative user experience and confirmed that the FPS drop is visually indistinguishable and that EVR delivers smooth user experiences. Although the goal of EVR is not to save bandwidth, EVR does reduce the network bandwidth requirement through SAS, which transmits only the pixels that fall within user's sight.

CONCLUSION

We anticipate that EVR will have a significant long-term impact on VR technologies and their applications of tomorrow. We summarize the main contributions as follows: EVR provides the first energy characterization study of VR devices and demonstrate their major energy overhead; EVR develops the first hardware accelerator for accelerating the critical PT operations in VR video processing; EVR provides a systematic approach to improve the energy efficiency of VR applications with cloud/client codesign.

Energy Characterization of VR Devices

Although there are many prior studies that focused on energy measurement of mobile devices, such as smartphones and smartwatches, this is the first such study that specifically focuses on VR devices. We show that the energy profiles of VR devices are significantly different from that of traditional mobile devices. Our results suggest that we must rethink the conventional system-level power/energy optimizations in the context of VR processing.

Although there are many prior studies that focused on energy measurement of mobile devices, such as smartphones and smartwatches, this is the first such study that specifically focuses on VR devices.

Implication on Hardware IP Block for VR

This article provides a case in point for future mobile SoCs to integrate VR-specific and VR-optimized IP blocks, and our principal idea of bypassing the GPU will be critical to those designs. We design the PTE as a standalone IP block in order to enable modularity and ease distribution. Alternatively, the PTE logic could be tightly integrated into either the video codec or display processor. Indeed, many new designs of the display processor have started integrating functionalities that used to be executed in GPUs, such as color space conversion. Such a tight integration would let the display processor directly perform PT operations before scanning out the frame to the display, and thus reduces the memory traffic induced by writing the FOV frames from the PTE to the frame buffer.

Cloud/Client Codesign for VR

EVR provides a cloud/client collaborative approach to improve the energy efficiency of VR devices. In EVR, SAS and HAR have different tradeoffs. On one hand, HAR is applicable regardless where the VR videos are from, but does not completely remove the overhead of the PT operation. SAS potentially removes the PT operation altogether, but relies on that the VR video is published to a cloud server first. We show that combining the two, when applicable, achieves the best energy efficiency.

REFERENCES

1. *WhitePaper: 360-Degree Video Rendering*. [Online]. Available: https://community.arm.com/graphics/b/blog/posts/white-paper-360-degreevideo-rendering
2. X. Chen, N. Ding, A. Jindal, C. Hu, M. Gupta, and R. Vannithamby, "Smartphone energy drain in the wild: Analysis and implications," *ACM SIGMETRICS Performance Eval. Rev.*, vol. 43, no. 1, pp. 151–164, 2015.
3. X. Corbillon, F. Simone, and G. Simon, "360-degree video head movement dataset," in *Proc. 8th ACM Multimedia Syst. Conf.*, 2017, pp. 199–204.
4. M. Halpern, Y. Zhu, and V. Reddi, "Mobile CPU's rise to power: Quantifying the impact of generational mobile CPU design trends on performance, energy, and user satisfaction," in *Proc. Int. Symp. High-Performance Comput. Archit.*, 2016, pp. 64–76.
5. B. Haynes, A. Minyaylov, M. Balazinska, L. Ceze, and A. Cheung, "VisualCloud demonstration: A DBMS for virtual reality," in *Proc. ACM Int. Conf. Manage. Data*, 2017, pp. 1615–1618.
6. J. Hegarty *et al.*, "Darkroom: Compiling high-level image processing code into hardware pipelines," in *Proc. SIGGRAPH*, 2014.
7. J. Huang, A. Badam, R. Chandra, and E. Nightingale, "WearDrive: Fast and energy-efficient storage for wearables," in *Proc. USENIX Annu. Tech. Conf.*, 2015, pp. 613–625.
8. K. Lee *et al.*, "Outatime: Using speculation to enable low-latency continuous interaction for mobile cloud gaming," in *Proc. 13th Annu. Int. Conf. Mobile Syst., Appl., Services*, 2015, pp. 151–165.
9. J. Li, A. Badam, R. Chandra, S. Swanson, B. Worthington, and Q. Zhang, "On the energy overhead of mobile storage systems," in *Proc. File Storage Technol.*, 2014, pp. 105–118.
10. F. Qian, L. Ji, Bo Han, and V. Gopalakrishnan, "Optimizing 360 video delivery over cellular networks," in *Proc. 5th Workshop All Things Cellular: Oper., Appl. Challenges*, 2016, pp. 1–6.

Yue Leng is currently a Software Engineer with Airbnb, San Francisco, CA, USA. Leng received the M.S. degree in computer engineering from the University of Illinois at Urbana–Champaign in 2019. Contact her at yueleng2@illinois.edu.

Jian Huang is currently an Assistant Professor with the Electrical and Computer Engineering Department, University of Illinois at Urbana-Champaign. Huang received the Ph.D. degree from Georgia Institute of Technology in 2017. Contact him at jianh@illinois.edu.

Chi-Chun Chen is currently a Compiler Engineer with Cray, Inc., Seattle, WA, USA. Chen received the M.S. degree in computer science from the University of Rochester in 2019. Contact him at cchen120@ur.rochester.edu.

Qiuyue Sun is currently a senior undergraduate student with the Computer Science Department, University of Rochester. Contact her at qsun15@u.rochester.edu.

Yuhao Zhu is currently an Assistant Professor with the Computer Science Department, University of Rochester. Zhu received the Ph.D. degree from The University of Texas at Austin in 2017. He is the corresponding author of this article. Contact him at yzhu@rochester.edu.