Q-VR: System-Level Design for Future Collaborative Virtual Reality Rendering

Chenhao Xie
Pacific Northwest National Lab

Xie Li
University of Sydney

Yang Hu
The University of Texas at Dallas

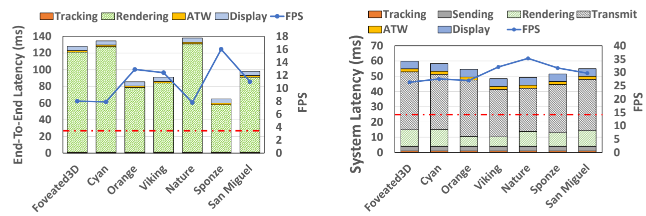
Huwan Peng
University of Washington

Michael B. Taylor
University of Washington

Shuaiwen Leon Song
University of Sydney and University of Washington

1. Motivation

Since the release of the movie *Ready Player One*, consumers have been longing for a commercial VR product which could provide a truly immersive experience without mobility restriction and periodical motion anomalies. In other words, users require exceptional visual quality from an *untethered* mobile-rendered head-mounted displays (HMDs) that is equivalent to what high-end tethered VR systems (e.g., Oculus Rift [15] and HTC Vive [7]) provide. Although the current mobile's processing capability has been significantly improved [1,16], they still cannot fully process heavy VR workloads under the stringent runtime latency constraints. With the development of high performance server technology, server-based realtime rendering of Computer Graphics has been introduced by the recent architecture studies [8, 10, 23, 24] and major cloud vendors [5, 14]. However, under the current network conditions, remote servers alone cannot provide realtime low-latency high-quality VR due to the dominating communication latency. Fig.1 shows the breakdown of the *end-to-end latency* (i.e., from tracking to display) for executing several high-quality VR applications under two commercial mobile VR designs: *local-only rendering* and *remote-only rendering*. The blue lines represent the frame rate (FPS) achieved on the VR HMD while the red dash lines illustrate VR system latency restriction (i.e., commercial standard of 25ms). The figure shows that the performance on the integrated GPU is the key bottleneck for local-only rendering, while the transmission latency in remote-only rendering contributes to approximately 63% of the overall system latency. Thus, neither local-only nor remote-only rendering can satisfy the latency requirements for high-quality mobile VR: there is a clear mismatch between hardware's raw computing power and desired rendering complexity.



(a) Local-only rendering (b) remote-only rendering

Figure 1: System latency and FPS when running high-end VR applications on two types of mobile VR system designs.

2. Limitations of the State of the Art

To address the latency and bandwidth challenges of today's dominant mobile rendering models, it seems reasonable to utilize mobile VR hardware's computing power to handle part of the rendering workload near the display HMD to trade off for reduced network communication, while letting the remote system handle the remaining workload. But how to design such VR systems to reach the latency and perception objectives is still an open problem. Recent studies [3, 9, 11–13] proposed a static collaborative software framework that renders the interactive objects locally while offloading the background environment to the remote servers. However, after a thorough qualitative investigation into its architecture-level rendering pipeline and a quantitative latency bottleneck analysis (Sec.2.3), we observe that this naive rendering scheme faces several challenges. First, interactive objects have to be narrowly defined by programmers on each hardware platform to satisfy the "worst case" scenario during VR development which significantly limits the design possibilities for high-quality interactive VR environments and burdens programmers to accommodate all the realtime constraints during the development cycle. It is labor intensive and impractical. Second, since the remote rendering workload remains unchanged, this scheme cannot drastically reduce the communication latency. Third, it loses the flexibility to dynamically maintain the balance between the local and remote rendering latency under realtime uncertainties: unpredictable user inputs (e.g., interaction and movements) and environment changes (e.g., hardware and network). Finally, it suffers from high composition overhead by requiring more complex collision detection and embedding methods [3, 9], directly contributing to resource contention on mobile GPU(s).

3. Key Insights

Unlike the previous pure software solutions, we explore how to build an efficient collaborative rendering pipeline for high-quality VR through a thorough workload characterization and a qualitative VR execution pipeline analysis (Sec.2.3). We summarize the following key design goals: (1) reducing the overall communication data size to decrease the global impact from remote rendering and transmission latency; (2) dynamically balancing local and remote rendering latency based on realtime constraints for optimal resource utilization and rendering efficiency; and (3) significantly reducing or even eliminating realtime hardware contention on the execution pipeline to further improve FPS. To reach these goals, we argue that the best strategy should be a *soft-hardware co-design solution* that transforms this complex VR execution problem into a cross-layer system design and optimization problem. To achieve (1), we propose a software framework design (Sec 3) based on two key insights: (a) human visual acuity falls off from the centre (called *fovea*) to the *periphery* [17, 19] and different acuity level requirements of human visual system naturally generate a new workload partitioning mechanism for collaborative VR rendering; (b) modern mobile SoCs are capable of dynamically rendering a range of workloads (or fovea sizes) with fine details and high resolutions, determined by realtime constraints. To achieve (2) and (3), we propose two novel hardware component designs. For (2), we discovered that there is a strong correlation between VR motion features and realtime hardware-level intermediate data, which can be leveraged to describe the scene complexity change and help dynamically build a strong mapping between environmental conditions and rendering workload (Sec.4.1). For (3), we observe that the major pipeline contention is caused by the resource competition on GPU(s) from the following concurrent executions: rendering, composition and asynchronous timewarp (ATW). We further discover that there is an algorithmic-level similarity between composition and ATW so that we can combine them through execution pipeline reordering to enable an asynchronous execution with GPU(s) for further improving the overall FPS (Sec.4.2).

4. Main Artifacts

In this paper, we propose a novel *software-hardware co-design for low-latency high-quality collaborative mobile VR*, named *Q-VR*, which effectively leverages the processing capability of both local and remote rendering hardware. At the software level, new interfaces and programming model are designed and integrated to Q-VR so that it can leverage the foveation effects of the human vision system [6, 18, 20–22] to enable a dynamic collaborative rendering framework for fine-grained rendering workload tuning and network latency reduction, while maintaining user perception (Section 3). The software-layer design also transforms this complex global collaborative rendering problem into a workable framework so that deeper hardware pipeline-level optimizations are possible. Specifically, at the hardware layer, based on the key insights above, we design a *lightweight interaction-aware workload controller* (Sec.4.1) and a *unified composition and reprojection unit* (Sec.4.2), to achieve two optimization objectives: (1) quickly reaching the local-remote latency balance for each frame to achieve the optimal rendering efficiency; and (2) further optimizing the global collaborative rendering pipeline for better architecture-level parallelism. To implement and evaluate the proposed Q-VR hardware design, we extend ATTILA-sim [2], a cycle-accurate rasterization-based GPU rendering simulator. Specifically, we implement simultaneous multi-projection engine in ATTILA-sim to support two-eyes VR rendering similar to [23] and reconfigure it by referencing the ARM Mali-G76 [4], a state-of-the-art high-end mobile GPU. Refer to Sec.5 for the detailed evaluation methodology.

5. Key Results and Contributions

As Sec.6.1 demonstrates, Q-VR achieves an average of **2.2x** (up to **3.1x**) end-to-end performance speedup and a **4.1x** frame rate improvement over the static collaborative rendering design. The runtime traces and sensitive study in Sec.6 show that Q-VR is able to help the system quickly reach local-remote balance under different user inputs and realtime environment constraints. Furthermore, Q-VR achieves an average of 73% energy reduction over the local rendering design. To summarize, the paper makes the following contributions:

- We design the first software-hardware co-designed collaborative rendering architecture to tackle the mismatch between VR hardware processing capability and desired rendering complexity from a cross-layer systematic perspective;
- We identify the fundamental limitations of the state-off-the-art collaborative rendering designs and quantify the major bottleneck factors via detailed workload characterization and VR execution pipeline analysis;
- By leveraging the foveation features of human visual system, we explore the software-level flexibility to reduce the network limitation via a fine-grained dynamic tuning space for workload control while maintaining user perception;
- Based on our key observations on VR motion correlations and execution similarity, we design two novel hardware components to support software-layer interfacing and deeper pipeline-level optimizations.

6. Why ASPLOS

Future collaborative mobile VR is a system design and optimization problem, requiring aggressive co-optimization of graphics algorithms, networking, vision, architecture and wireless. Necessarily there is no single expert on every aspect of it; in this paper we capture the current system level constraints and see what role jointly modifying the hardware and software can play, which makes perfect synergy with ASPLOS's long efforts on multidisciplinary ground-breaking research. From the the perspective of research scope, Q-VR is a quintessential ASPLOS paper, representing the intersection between software-hardware co-design and emerging applications. It also includes case studies of real-world experimental systems.

7. Citation for Most Influential Paper Award

In this work, authors propose the first software-hardware co-designed collaborative rendering architecture to tackle the mismatch between VR hardware processing capability and desired rendering complexity, from a cross-layer systematic perspective. Their solution has provided a possible pathway to design future low-latency high-quality mobile VR systems and paves the road for other research in computer system/architecture to be proposed for this very important area.

References

[1] Apple. Iphone 11 pro. https://www.apple.com/iphone-11-pro/, 2018.

[2] Victor Moya Del Barrio, Carlos González, Jordi Roca, Agustin Fernández, and Roger Espasa. Attila: a cycle-level execution-driven simulator for modern gpu architectures. *2006 IEEE International Symposium on Performance Analysis of Systems and Software*, pages 231–241, 2006.

[3] Kevin Boos, David Chu, and Eduardo Cuervo. Flashback: Immersive virtual reality on mobile devices via rendering memoization. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '16, pages 291–304, New York, NY, USA, 2016. ACM.

[4] ARM Developer. Mali-g76 high performance gpu. https://developer.arm.com/ip-products/graphics-and-multimedia/mali-gpus/mali-g76-gpu, 2017.

[5] Google. Google Cloud for games. https://cloud.google.com/solutions/gaming/, 2017.

[6] Brian Guenter, Mark Finch, Steven Drucker, Desney Tan, and John Snyder. Foveated 3d graphics. *ACM Trans. Graph.*, 31(6):164:1–164:10, November 2012.

[7] HTC-Vive. Vive VR products. https://www.vive.com/us/, 2019.

[8] Youngsok Kim, Jae-Eon Jo, Hanhwi Jang, Minsoo Rhu, Hanjun Kim, and Jangwoo Kim. Gpupd: a fast and scalable multi-gpu architecture using cooperative projection and distribution. In *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 574–586, 2017.

[9] Zeqi Lai, Y. Charlie Hu, Yong Cui, Linhui Sun, and Ningwei Dai. Furion: Engineering high-quality immersive virtual reality on today's mobile devices. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*, MobiCom '17, pages 409–421, New York, NY, USA, 2017. ACM.

[10] Yue Leng, Chi-Chun Chen, Qiuyue Sun, Jian Huang, and Yuhao Zhu. Energy-efficient video processing for virtual reality. In *Proceedings of the 46th International Symposium on Computer Architecture*, pages 91–103, 2019.

[11] Xing Liu, Christina Vlachou, Feng Qian, Chendong Wang, and Kyu-Han Kim. Firefly: Untethered multi-user VR for commodity mobile devices. In *2020 USENIX Annual Technical Conference (USENIX ATC 20)*, pages 943–957. USENIX Association, July 2020.

[12] Simone Mangiante, Guenter Klas, Amit Navon, Zhuang GuanHua, Ju Ran, and Marco Dias Silva. Vr is on the edge: How to deliver 360 degree videos in mobile networks. In *Proceedings of the Workshop on Virtual Reality and Augmented Reality Network*, VR/AR Network '17, pages 30–35, New York, NY, USA, 2017. ACM.

[13] Jiayi Meng, Sibendu Paul, and Y. Charlie Hu. Coterie: Exploiting frame similarity to enable high-quality multiplayer vr on commodity mobile devices. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '20, page 923–937, New York, NY, USA, 2020. Association for Computing Machinery.

[14] Nvidia. Geforce Now. https://www.nvidia.com/en-gb/geforce/products/geforce-now/, 2018.

[15] Oculus. Oculus VR products. https://www.oculus.com/, 2018.

[16] Qualcomm. snapdragon 855+mobile platform. https://www.qualcomm.com/products/snapdragon-855-plus-mobile-platform, 2018.

[17] Ruth Rosenholtz. Capabilities and limitations of peripheral vision. *Annual Review of Vision Science*, 2:437–457, 2016.

[18] Michael Stengel, Steve Grogorick, Martin Eisemann, and Marcus Magnor. Adaptive image-space sampling for gaze-contingent real-time rendering. In *Computer Graphics Forum*, volume 35, pages 129–139. Wiley Online Library, 2016.

[19] Hans Strasburger, Ingo Rentschler, and Martin Jüttner. Peripheral vision and pattern recognition: A review. *Journal of vision*, 11(5):13–13, 2011.

[20] Zhou Wang, Alan C Bovik, and Ligang Lu. Wavelet-based foveated image quality measurement for region of interest image coding. In *Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205)*, volume 2, pages 89–92. IEEE, 2001.

[21] Zhou Wang, Alan Conrad Bovik, Ligang Lu, and Jack L Kouloheris. Foveated wavelet image quality index. In *Applications of Digital Image Processing XXIV*, volume 4472, pages 42–52. International Society for Optics and Photonics, 2001.

[22] Martin Weier, Thorsten Roth, Ernst Kruijff, André Hinkenjann, Arsène Pérard-Gayot, Philipp Slusallek, and Yongmin Li. Foveated real-time ray tracing for head-mounted displays. In *Computer Graphics Forum*, volume 35, pages 289–298. Wiley Online Library, 2016.

[23] Chenhao Xie, Fu Xin, Mingsong Chen, and Shuaiwen Leon Song. Oo-vr: Numa friendly object-oriented vr rendering framework for future numa-based multi-gpu systems. In *Proceedings of the 46th International Symposium on Computer Architecture*, ISCA '19, pages 53–65, New York, NY, USA, 2019. ACM.

[24] Chenhao Xie, Xingyao Zhang, Ang Li, Xin Fu, and Shuaiwen Leon Song. Pim-vr: Erasing motion anomalies in highly-interactive virtual reality world with customized memory cube. In *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, Feb 2019.