Matthew Monsour

Prof. Southworth

CSE 300 Technical Communications

20 April 2016

Vulkan: High Performance Computer Graphics of the Future

Thesis:

The Vulkan API offers a new way forward for high performance graphics across a wide variety of computer platforms, solving the many problems of its predecessors.

Computer graphics APIs (Application Programming Interface) have been evolving for over 20 years, with each new iteration providing its own improvements for computer graphics software to take advantage of. To clearly define what a graphics API is, one can read the introduction to the OpenGL specification as a concrete example. “OpenGL is a set of commands that control the operation of the GPU. Modern GPUs accelerate almost all OpenGL operations,” the specification reads (Segal 2). To clarify, a graphics API is a set of commands that allows a programmer to control the graphics processor, which can be used to accelerate calculations that are important in computer graphics. OpenGL is a notable example, as it is considered a predecessor to the Vulkan API (“Khronos”).

OpenGL was first released in 1992 as a graphics API meant to allow programmers to more easily take advantage of GPU features. Since its initial release, a number of major releases have included both improvements and revisions to the original specification. In 2008, revisions that would remove certain API features were released, as these features no longer reflected the design of modern graphics hardware. Aside from the issues with aging, a number of other issues in computer graphics began to appear, creating a need for a newly designed graphics API that could elegantly solve these issues.

A major flaw with OpenGL's design was its focus on desktop computer platforms. As other platforms gained popularity and a need for computer graphics software, they would also need graphics APIs. Unfortunately, OpenGL was not suitable for use on many of these platforms, such as embedded systems. On early embedded systems, some of the most important hardware that OpenGL was designed to take advantage of was widely unavailable (e.g. FPUs for fast floating point calculation) (Pulli 43). In order to deal with the differences in these platforms, Khronos Group wrote a specification for OpenGL ES, a graphics API specifically for embedded

systems (Pulli 43). This would mean that devices such as heads up displays for vehicles, or even mobile phones would then have access to graphics APIs comparable to the OpenGL standard on desktop (Baek 166).

Despite the design influence from OpenGL, OpenGL ES does not remain perfectly compatible with any version of OpenGL, and the two APIs began to evolve separately (Baek 167). As many embedded systems now closely resemble desktop graphics hardware, the distinction between these APIs is no longer strictly necessary. And, in fact, now neither API closely resembles the actual design of the hardware it is implemented on top of. This results in a loss of performance, as abstractions over the hardware require the software emulation of features provided by the API, but not the hardware.

One last major issue with OpenGL is the performance impact of driver software. Drivers provide access to hardware functionality, and usually implement high level features of graphics APIs that are not necessarily supported by the graphics hardware itself. Drivers exist largely outside of the control of graphics software programmers, so any delays in the driver are out of the control of a graphics programmer using OpenGL as well. Due to this, using OpenGL creates difficulties when trying to write high performance computer graphics applications (Olson 31).

Because of all the issues in computer graphics that have accumulated over the years, Khronos Group began the design of Vulkan, intending to correct many of the common issues in computer graphics software, as well as to provide a new solution more suitable for high performance graphics software. Vulkan solves the problem of fragmentation by providing a single API for all major platforms (“Khronos”). Where OpenGL and OpenGL ES provided separate APIs for different groups of platforms, Vulkan aims to unify both groups of platforms. As of its release, Vulkan supports both desktop and mobile platforms (“Khronos”). In addition to

improving support across different platforms, Vulkan also aims to reduce unnecessary time spent in driver software (“Khronos”). While there are drawbacks to both of these design decisions, ultimately the API provides a much better solution for high performance computer graphics than any of its predecessors. Since Vulkan was officially released only in February of 2016, not much data is available on its performance relative to OpenGL, OpenGL ES, or any other comparable APIs. However, in the near future we are likely to see the benefits that come with Vulkan, and research that demonstrates the improvements promised to come with its ground up design.

Over many years, I have followed developments in the field of computer graphics. While I would say I have an amateur understanding of many topics in the field, I still manage to find a lot of interesting reading in this area. The topic of Vulkan came to my attention upon its announcement last year. Having read a lot of articles about the issues that come with OpenGL and similar APIs, as well as some of the early research into high performance graphics APIs with projects like Mantle, I was naturally interested in the new direction of computer graphics. Ultimately, Vulkan became my primary topic of interest due to its very recent public release. In the future, I expect many of the ideas that Vulkan was designed around to have a great impact on the field of computer graphics.

Doing research on the state of modern computer graphics was very interesting as well, which made the topic an easy choice for this paper. Doing further research into the history of computer graphics, as well as the potential of modern approaches to the problems faced by early graphics APIs make up a significant part of this paper. Since the field of computer graphics is so diverse, but features many very focused topics, doing research into this area proved to be very rewarding. Due to both my initial interest in this topic, and the interest I gained from doing further research into the topic, Vulkan and high performance computer graphics became a very good choice for the topic of this paper. As such, I decided early on that this was the topic I would write about for this paper.

Works Cited

"Khronos Releases Vulkan 1.0 Specification." *Khronos Group*. Khronos Group, 16 February 2016. Web. 24 February 2016.

Segal, Mark, and Kurt Akeley. "The OpenGL Graphics System: A Specification." Khronos Group, 28 May 2015. PDF file.

Olson, Tom. "Vulkan: The future of high-performance graphics." Khronos Group, 2014. PDF file.

Baek, N., and K.-H. Yoo. "Emulating OpenGL ES 2.0 Over the Desktop OpenGL." *Cluster Computing* 18.1 (2015): 165-175. *Scopus*®. Web. 24 Feb. 2016.

Pulli, Kari1, Jani Vaarala, and Ville Miettinen. "Refining OpenGL for Embedded Systems." *ACM Queue* (2007): 42-43. *Applied Science & Technology Source*. Web. 24 Feb. 2016.