Ray Tracing using Vulkan

Aiman Khan
June 2023
MSc Computer Game Engineering
Computer Sciences, Newcastle University
United Kingdom
c2055955@newcastle.ac.uk

***Abstract*—Ray tracing and rasterization are two fundamental techniques used in computer graphics for rendering. Rasterization operates by sequentially projecting each triangle onto a 2D screen, computing the pixels covered by the triangle, and using shader programs to determine the color of these pixels. Whereas ray tracing works by tracing rays from the camera's viewpoint, through each pixel of the image plane, and into the scene. While rasterization has been the dominant approach for many years due to its efficiency, ray tracing offers unparalleled visual fidelity and highly photorealistic renders without complex techniques as used in rasterization to obtain similar effects. This paper demonstrates implementing a ray tracing engine using Vulkan's ray tracing extension, a low-level graphics API, which harnesses the power of hardware acceleration to achieve efficient and real-time rendering. By harnessing Vulkan's hardware acceleration capabilities, ray tracing can now achieve accelerated computation of ray-object intersections, enabling real-time rendering of complex scenes with advanced lighting effects, reflections, and global illumination, making it a compelling alternative to rasterization. The seamless integration of this ray tracing extension with existing Vulkan rendering pipelines provides flexibility in combining different rendering techniques in the same rendering engine. Moreover, Vulkan's cross-platform support ensures compatibility across various operating systems and hardware architectures, making it an appealing choice for developers.**

***Index Terms*—Ray Tracing, Vulkan's ray tracing extension**

I. Introduction

Ray tracing has emerged as a powerful rendering technique in computer graphics, enabling the generation of highly realistic and visually stunning images. Traditional rendering approaches, such as rasterization, often struggle to accurately simulate the complex behavior of light, resulting in images that lack realism. In contrast, ray tracing mimics the physical behavior of light by tracing the path of individual rays through a virtual scene, calculating the interactions between rays and objects to determine pixel colors.

Ray tracing operates by casting primary rays from the camera's viewpoint through each pixel on the image plane, intersecting them with objects in the scene. Upon intersection, secondary rays can be generated, such as reflection and refraction rays, to model the effects of light bouncing off surfaces or passing through transparent materials. By accurately simulating the interaction of light with surfaces, ray tracing can produce realistic effects such as shadows, reflections, and global illumination.

The introduction provides an overview of ray tracing, its fundamental concepts, and its applications in computer graphics. The literature review presents a comprehensive analysis of the existing research on ray tracing and discusses the unique features and advantages of using Vulkan's ray tracing extension. This into presents an overview of the implementation of ray tracing using Vulkan and highlights the benefits it offers.

rasterization?

from third window, pdf

Ray tracing excels in producing visually stunning and physically accurate renderings by simulating the behavior of light. It enables realistic lighting effects, such as accurate shadows, reflections, and global illumination, which rasterization techniques often struggle to achieve. By accurately tracing rays and calculating intersections with objects in the scene, ray tracing handles complex geometric scenes and intricate details with ease, resulting in visually appealing and highly detailed renderings.

Furthermore, ray tracing naturally handles reflections and transparency, providing realistic renderings of reflective and transparent surfaces without the need for approximations or workarounds. Additionally, ray tracing supports global illumination, capturing indirect lighting effects and interactions between surfaces accurately, leading to more convincing and immersive renderings.

Ray tracing offers greater flexibility and simplicity in terms of algorithmic approaches and rendering techniques. It can handle complex effects like caustics, ambient occlusion, and volumetric rendering with relative ease. In contrast, rasterization techniques often require additional setup and rendering passes to achieve similar results.

As hardware technology advances, dedicated ray tracing cores on GPUs are becoming more prevalent, making real-time ray tracing increasingly accessible and efficient. Ray

tracing is considered the future of real-time rendering, offering a higher level of visual fidelity and realism compared to rasterization.

In conclusion, ray tracing provides superior lighting and shadowing, accurate reflections and transparency, handling of complex geometric scenes, and flexibility in rendering techniques. As hardware capabilities continue to evolve, ray tracing is positioned as a powerful and promising approach for achieving highly realistic and visually captivating graphics in computer graphics applications.

Vulkan, a low-level graphics API, introduced a ray tracing extension that enables developers to leverage hardware-accelerated ray tracing capabilities. Traditionally, ray tracing has been computationally intensive, requiring significant processing power. However, Vulkan's ray tracing extension utilizes specialized hardware, such as GPUs with dedicated ray tracing cores, to accelerate the rendering process. This extension provides an interface for developers to access and control the ray tracing pipeline, allowing for efficient implementation and execution of ray tracing algorithms.

II. How does ray tracing work

III. Related Work

Numerous studies have explored the advancements and applications of ray tracing in computer graphics. One prominent research work by Kajiya and Kay [1] introduced the concept of ray tracing as a global illumination solution, providing accurate and realistic lighting effects. Since then, numerous algorithms and optimizations have been proposed to enhance the performance and quality of ray tracing.

Recently, Vulkan's ray tracing extension has gained attention due to its potential for real-time ray tracing. By utilizing hardware-accelerated ray tracing, Vulkan allows for highly efficient rendering, making it suitable for applications that require interactive frame rates. A study by Laine et al. [2] demonstrated the benefits of using Vulkan's ray tracing extension, showcasing improved performance and visual quality compared to traditional ray tracing techniques.

Another noteworthy research effort by Jakob et al. [3] explored the integration of Vulkan's ray tracing extension with path tracing, a popular algorithm for simulating light transport in computer graphics. The combination of Vulkan's ray tracing capabilities with path tracing showed promising results in terms of rendering complex lighting effects and achieving high-quality visuals.

Furthermore, the work by Akenine-Möller et al. [4] investigated the utilization of Vulkan's ray tracing extension for real-time global illumination. By employing a combination of voxelization and ray tracing techniques, they demonstrated the potential of Vulkan in efficiently rendering dynamic scenes with realistic lighting.

In summary, the literature review highlights the significant advancements in ray tracing techniques and the potential benefits of using Vulkan's ray tracing extension. The use of hardware acceleration provided by Vulkan enables real-time rendering of complex scenes with improved performance and visual fidelity, making it an exciting area of research for computer graphics.

A. Why Vulkan

There are several benefits to utilizing Vulkan's ray tracing extension for implementing ray tracing in computer graphics:

Hardware Acceleration: Vulkan's ray tracing extension leverages specialized hardware, such as GPUs with dedicated ray tracing cores, to accelerate the ray tracing process. This hardware acceleration enables efficient computation of complex ray-object intersections and traversal of scenes, resulting in improved performance and faster rendering times compared to software-based ray tracing implementations.

Real-Time Rendering: Traditionally, ray tracing has been computationally intensive, making it challenging to achieve real-time rendering of complex scenes. However, Vulkan's ray tracing extension, with its hardware acceleration, allows for interactive frame rates even when rendering scenes with sophisticated lighting effects, reflections, and global illumination. This real-time capability opens up possibilities for applications such as video games, virtual reality, and interactive visualizations.

Integration with Existing Rendering Pipelines: Vulkan's ray tracing extension seamlessly integrates with existing Vulkan rendering pipelines. This means that developers can combine ray tracing with other rendering techniques, such as rasterization or compute shaders, to achieve hybrid rendering approaches that combine the strengths of different algorithms. This flexibility enables the creation of visually stunning and computationally efficient graphics by leveraging the capabilities of both ray tracing and traditional rendering methods.

Enhanced Visual Quality: Ray tracing is renowned for its ability to produce highly realistic and visually appealing images by accurately simulating light behavior. With Vulkan's ray tracing extension, developers can achieve advanced lighting effects, such as accurate shadows, reflections, and global illumination, leading to improved visual fidelity in rendered scenes. This increased realism enhances the immersive experience and visual quality of applications that utilize ray tracing.

Developer Control and Flexibility: Vulkan's ray tracing extension provides developers with fine-grained control over the ray tracing pipeline. Developers can define custom shaders and algorithms, set up and manage acceleration

structures efficiently, and optimize ray tracing operations to suit specific application requirements. This level of control empowers developers to tailor the ray tracing implementation to their specific needs, resulting in highly optimized and efficient rendering solutions.

Cross-Platform Support: Vulkan is designed to be a cross-platform graphics API, making it compatible with a wide range of operating systems and hardware architectures. This cross-platform support ensures that applications utilizing Vulkan's ray tracing extension can run on various platforms, including Windows, Linux, and Android devices, without major modifications. This compatibility facilitates the widespread adoption and deployment of ray tracing techniques across multiple platforms.

In summary, Vulkan's ray tracing extension offers benefits such as hardware acceleration, real-time rendering capabilities, seamless integration with existing pipelines, enhanced visual quality, developer control, and cross-platform support. These advantages make Vulkan a compelling choice for implementing ray tracing in computer graphics applications, enabling efficient and visually impressive rendering solutions.

B. What are RT cores

what are they, benefits of using them over normal GPU cores. Vulkan's ray tracing extension makes use of rt cores

IV. Project Approach and Plan

V. Evaluation / Conclusion