Visual Raytrace: An Immersive Learning Application

Manfred Brill, Benedict Särota

University of Applied Sciences Kaiserslautern
Amerikastr. 1
66482 Zweibrücken
Tel.: +49 (0)631 / 37 24 - [5382, 5321]
E-Mail: [manfred.brill, benedict.saerota]@hs-kl.de

Abstract: Visual Raytrace is an immersive learning application supporting students in a computer graphics class to create their own version of a working raytracing software. Users interact with the ingredients of a raytracer like coordinate systems, cameras, framebuffer, scene description, sampling, reflection models or texture mapping.

Keywords: Virtual Reality, Immersive Learning, Computer Graphics, Raytracing

1 Introduction

Raytracing, introduced by Turner Whitted [Whi80a], is one of the major topics in computer graphics classes. In figure 1 we find one of the famous pictures rendered by Whitted.

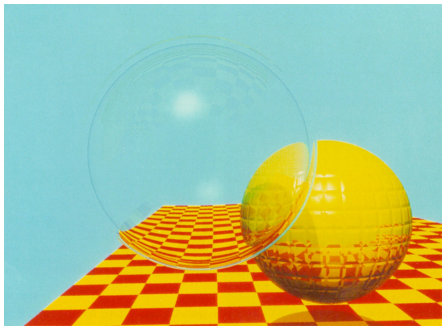

Figure 1: Image rendered by Turner Whitted [Whi80b]

Usually students in a computer graphics class implement their version of such a renderer, after getting the theory in the class. To achieve this they need to understand the basic concepts of computer graphics like coordinate systems, cameras, reflection models or texture mapping. To implement a raytracer or another 3D application the students have to develop a strong visual-spatial ability. Virtual reality applications provide joy of use and support the transfer from 3D space to a programming language and deepen the understanding of the basic concepts of a raytracer.

2 Visual Raytrace

We implemented the immersive learning application **Visual Raytrace** [Sä21a, Sä21b] using the Unity Game Engine and HTC Vive Input Utility [Won21]. Figure 2 shows the main features of the application. The scene to be rendered is placed in the virtual environment as a world in a miniature [SCP95].

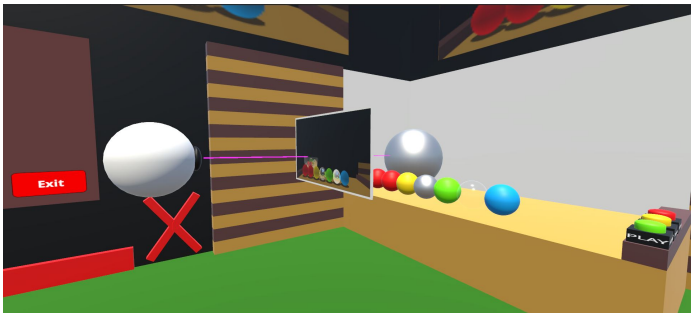

Figure 2: The elements of a raytracer in an immersive environment

The raytracer is implemented in C# and can be slowed down to make sure the users can follow the path of the ray and the reflections in the scene until the color of the fragment is finally computed. The raytracing process can be paused or stopped to get an understanding of the rendering algorithm. At the end the computed image can be saved for further investigation.

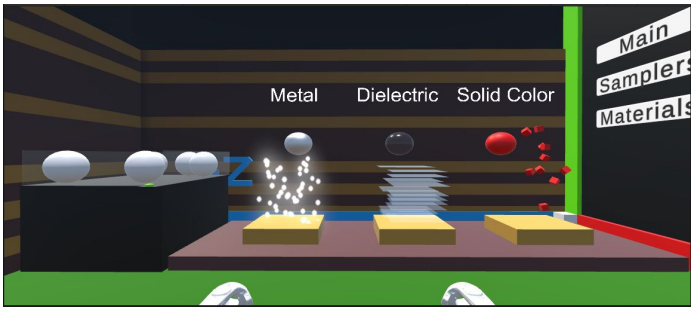

Figure 3: Geometric objects and reflection functions for scene definition

The raytracing scene can either be defined in VR or can be read from a scene description file. Figure 3 shows the interface for the interactive scene definition. New objects can be picked, one of the reflection models can be assigned and then the object is positioned in the raytracing scene. The options for the raytracer can also be changed in the immersive environment.

3 Future Work

It is planned to implement a change of scenes so the users can dive into the rendered 3D world. This way users could walk around in the rendered scene and inspect intersection

points, reflections, refractions or other details. We tested the application on HTC Vive Pro and Vive Focus Plus. We work on builds for Google Cardboard, Oculus Android, OpenXR and WebXR to support as many platforms as possible.

Due to the pandemic situation on our campus we could not evaluate the application with students. Hopefully we will be able to do this in the winter term 2021/22 at our campus. The results of this evaluation will help us to improve the usability and joy of use of the application.

We plan to use the immersive learning application in the next computer graphics class in our department in summer 2022. The VR application will open up new ways of teaching [GC08, VGV+20]. To support this we will transfer the C# code for the raytracer to an own repository, so the raytracing code can be used stand-alone.

References

[GC08] Fabio Ganovelli and Massimiliano Corsini. eNVyMyCar: a Multi-player Car Racing Game for Teaching Computer Graphics. In Steve Cunningham and Lars Kjelldahl, editors, *Eurographics 2008 - Education Papers*. The Eurographics Association, 2008.

[Sä21a] Benedict Särota. Implementation of a vr application for the visualization of a raytracing process, 2021. Project Work, University of Applied Sciences Kaiserslautern.

[Sä21b] Benedict Särota. Raytracing. `https://github.com/VRLAB-HSKL/RayTracing`, 2021. Last Access: July 22., 2021.

[SCP95] Stoakley, Richard, Conway, Matthew, and Pausch, Randy. Virtual reality on a wim: Interactive worlds in miniature. In *Proceedings of CHI95*, pages 265 – 272. ACM, 1995.

[VGV+20] Nick Vitsas, Anastasios Gkaravelis, Andreas-Alexandros Vasilakis, Konstantinos Vardis, and Georgios Papaioannou. Rayground: An Online Educational Tool for Ray Tracing. In Mario Romero and Beatrice Sousa Santos, editors, *Eurographics 2020 - Education Papers*. The Eurographics Association, 2020.

[Whi80a] Turner Whitted. An improved illumination model for shaded display. *Communications of the ACM*, 23(6):343 – 349, 1980.

[Whi80b] Turner Whitted. Spheres and checkerboard. `https://de.wikipedia.org/wiki/Datei:Spheres_and_Checkerboard_-_Turner_Whitted.jpg`, 1980. Last Access: July 22., 2021.

[Won21] Lawrence Wong. Vive input utility for unity. `https://github.com/ViveSoftware/ViveInputUtility-Unity`, 2021. Last Access: July 22., 2021.