Virtual Optical Bench: A VR learning tool for optical design

Sebastian Pape* Visual Computing Institute
Martin Bellgardt* Visual Computing Institute
David Gilbert* Visual Computing Institute
Georg König** Chair for Technology of Optical Systems
Torsten W. Kuhlen* Visual Computing Institute
RWTH Aachen University, Germany

Abstract

The design of optical lens assemblies is a difficult process that requires lots of expertise. The teaching of this process today is done on physical optical benches, which are often too expensive for students to purchase. One way of circumventing these costs is to use software to simulate the optical bench. This work presents a virtual optical bench, which leverages realtime ray tracing in combination with VR rendering to create a teaching tool which creates a repeatable, non-hazardous and feature-rich learning environment. The resulting application was evaluated in an expert review with 6 optical engineers.

Index Terms: Applied computing—Education—Interactive learning environments——Applied computing—Physical sciences and engineering—Physics Computing methodologies—Computer graphics—Rendering—Ray tracing

1 Introduction

In today's technology, optical lens systems are getting more important with the dissemination of cameras, telescopes, and lasers in more parts of daily life. These lens systems are often stacked glass or plastic lenses aligned along an optical axis encapsulated in a housing to hold them in the desired distance to each other. The classical way of testing new lens assemblies is a physical optical bench. These tables or table tops are often located in clean rooms when operated with high power lasers, thus resulting in a high operation cost. The biggest downside of this method is the time spent in overhead to test simple adjustments and the necessity of owning the right lenses to test. The resulting initial costs for the bench and for a kit of spherical lenses and mounts to hold the lenses are too high to be purchased by students for their studies. One obvious way of circumventing these downsides is to use software for computer aided lens design. One of the tools widely used in the industry to simulate lens assemblies is OpticStudio by Zemax [12], which uses ray tracing to simulate laser patterns or light falling through the lens assemblies. Due to its focus on professional users it is well suited for testing lens assemblies, but not that well suited for students that want to learn about lens design and the optical effects.

2 Related Work

To this day there are multiple free tools that allow the simulation of optical effects in a 2D visualization.
Notable free tools are the Optical Table Simulator [10] and the OpticalRayTracer [5]. They both provide applets that allow the user to define virtual scenes with different optical elements. Furthermore, Tantalum [1] is a WebGL based application, which performs light tracing live in the browser. It is not aimed at professionals or optics hobbyists due to its predefined scenes which do not offer any manipulation of the optical elements except the light caster.

*{pape|bellgardt|gilbert|kuhlen}@vr.rwth-aachen.de
**georg.koenig@tos.rwth-aachen.de

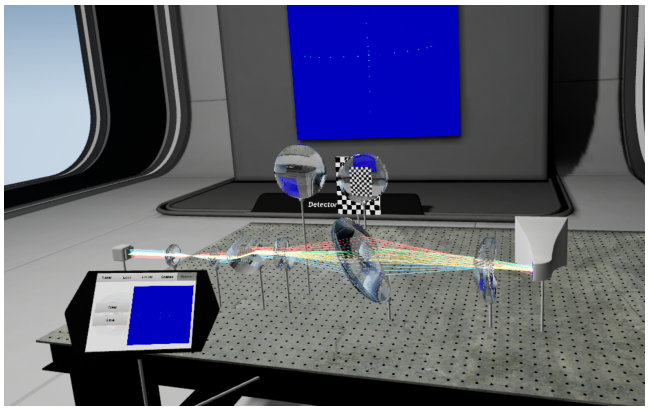

Figure 1: Our virtual optical bench application. Further material can be found at: https://vr.rwth-aachen.de/publication/02199/

At the current time, to the authors' knowledge, only one working 3D (or VR) optical bench application [11] exists. On the first impression it offers a lot of similar features. On closer inspection the available features are inaccurate polygonal approximation of a few lens types, while the simulated lasers are not adjustable in any specifications.
In addition to the free tools presented above, industry tools like OpticStudio by Zemax [12] offer many features, but are not well suited for teaching due to their focus on professional users and cost.
While the 2D set of tools would allow the usage in a classroom, we aimed at the usage of virtual reality in combination with realtime ray tracing for the 3D manipulation and creation of optical assemblies.
In addition to the presented tools, two papers are related to the work presented here. Kolb et al. [4] researched the simulation of a real camera in a ray tracing application with a virtual image sensor and a multi lens setup. Here they studied the effects of virtual lenses on the virtual camera and the calculations needed to bring different objects into focus. While this seems closely related to this work, only small optimizations from the paper could be used for our developments.
Another approach from Heinrich et al. [3] shows a method for simulating the effects of lenses on the resulting image in interactive applications. The authors derive the lenses' transformations and apply these to the geometry. This results in much higher frame rates than the ray tracing approach from [4], but can't deal with lenses that are placed off the center axis of the camera.

3 Technology

While the first implementation of this work was done in a self developed framework, a reimplementation was done in the Unreal Engine, see Figure 1.
To support ray tracing at the time of implementation, Nvidia OptiX [6] was integrated into the engine. This was necessary since Unreal did not support ray tracing in VR and had no support for implicit geometry. To realize the interaction between OptiX and Unreal, proxy objects with associated OptiX resources were created, which are rendered to a CUDA buffer and then copied into an Unreal/DirectX texture buffer.

In the following, the different components and their respective functions are described:

Camera As the first object, a backwards ray tracing camera was implemented [9]. Here "Vision Rays" are cast into the scene opposed to the physical light transport that would occur in the reverse direction. While this optimization only offers limited physical correctness, it still enables a decent simulation of optical effects for the user, while expensive path tracing is avoided. To further speed up the rendering, only the optical elements and the laser detector are rendered by the ray tracing camera. The ray tracing camera and the Unreal camera are moved in sync and the resulting buffers are combined to gain a complete picture.

Laser To evaluate lens systems in real setups, often laser emitters with different patterns are used. To simulate this, forward ray tracing is employed. Here infinitely thin light rays are traced from the laser through all optical elements in the scene. To simulate a variable thickness of the beam without lowering performance, the starting position of the traced light ray is altered according to a normal distribution multiple times per frame. To generate the normal distribution on the graphics card the *Box-Muller* transformation [2] is applied to a uniform pseudo random distribution. The resulting traced laser lines are then rendered as billboard geometry in the engine. Additionally, the patterns of the laser and its uniform wave length can be adjusted.

Detector To visualize the distribution patterns of the laser, a laser detector is implemented. The laser detector can be placed anywhere in the optical assembly and visualizes a 2D slice of the current laser beams with regard to the lasers' thickness. If the detection surface of the detector is placed into the laser beam the detected laser points are visualized as a 2D normalized histogram with a coloring similar to industrial infrared camera sensors on its surfaces. The visualization plane is tilted 90° to its detection plane, since this is the most likely angle from which the viewers would look at the detector.

Optical Elements As optical elements, spherical and planar lenses as well as planar textured objects are implemented. These lenses are defined as implicit geometry to accelerate ray tracing calculations and the ability to change all their parameters at runtime. The OptiX framework modularized all elements, making them easily extensible. While the planar targets simply stop the laser beam, the lenses bend the beam. The glass material simulation consists of the Fresnel-Schlick approximation [7] in combination with the *Sellmeier* [8] equation do determine the refraction index for different wavelengths of the same glass. Currently, 6 different glass types are predefined which represent a set often used by the end users. A more complicated glass simulation was tested but was not found more useful by the end users in first tests.

4 User Interface

The resulting application consists of a room containing a 3D model of an optical bench. Any desired amount of passive optical elements can be added to the bench. The only predefined elements are a laser emitter, the laser detector and a virtual handheld tablet.
With the tablet as a central control hub, the user can change the parameters of all components or create new components. All components can then be placed anywhere on the table. Since the optical elements are rendered by the ray tracing camera, they also provide a visual deflection of light rays for the eye of the user. This feature is only partially physically accurate, since backward ray tracing is employed. Fortunately this approximation still allows for the experimentation with the optical properties of the lenses and the construction of e.g. a zooming lens setup.
In the construction of the lens setup the 6-DoF controllers of the HMD are used for input. Since the usual tracking of such devices might not be sufficient for the precise alignment of the lens setups, multiple methods are used to increase precision. To align all elements on a straight line, snap lines are placed on the virtual bench for precisely snapping optical elements into alignment.
Additionally, two transformation gizmos are implemented. The first gizmo behaves similar to the transposing handles of most 3D design software, but the controller movement is scaled down by a factor of 10 and the handles stretch to accommodate for the scale down. In our tests the scaling factor of 10 resulted in a good balance between usability and precision.
The second gizmo is used for rotation. This gizmo is specialized on the rotation used in this application. It does not *roll* the objects, since these are mostly rotationally symmetric. The gizmo consists of two rings which are centered around the object and are textured with a degree-scale on them. From the object two manipulation handles stick out through the rings, such that a rough reading on the scale is possible. These handles are also stretchable, such that the user can extend them as much as needed to increase the angular precision. This results in arbitrary precision in angular placements. To increase the user perception of the entered numerical values, a floating label is placed on the end of each handle containing the value.

5 Conclusion and Future Work

So far the original implementation was evaluated in 6 think-aloud expert-review sessions with employees of the Chair for Technology of Optical Systems. The suggestions were taken into account for the reimplementation and many of them lead to improvements. Overall the application was deemed helpful by the experts to roughly construct optical systems and all of them agreed that it would help students to gain more intuitive understanding. In addition to the main question, all users stated that they were positively impressed by the possibilities that a simulation could offer them.
In the future this tool will be used in a lecture given at the university. To further evaluate the usefulness for teaching, the students' learning performance will be evaluated to expand this work.

Acknowledgments

The authors wish to thank all the testers from the Chair for Technology of Optical Systems. This work is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC 2023 Internet of Production – 390621612

References

[1] B. Bitterli. Tantalum. https://benedikt-bitterli.me/tantalum/, Oct 2015. [Online, accessed 13.10.2020].
[2] G. Box, M. E. Muller, et al. A note on the generation of random normal deviates. *Ann. Math. Stat.*, 29(2):610–611, 1958.
[3] W. Heidrich, P. Slusallek, and H.-P. Seidel. An image-based model for realistic lens systems in interactive computer graphics. In *GI*, p. 68–75. CIPS, 1997.
[4] C. Kolb, D. Mitchell, and P. Hanrahan. A realistic camera model for computer graphics. In *SIGGRAPH*, p. 317–324. ACM, 1995.
[5] P. Lutus. OpticalRayTracer. https://arachnoid.com/OpticalRayTracer/, Dec 2017. [Online, accessed 13.10.2020].
[6] S. G. Parker, J. Bigler, et al. Optix: a general purpose ray tracing engine. *TOG*, 29(4), 2010.
[7] C. Schlick. An inexpensive BRDF model for physically-based rendering. In *Computer graphics forum*, vol. 13, pp. 233–246. Wiley Online Library, 1994.
[8] W. Sellmeier. Zur Erklärung der abnormen Farbenfolge im Spectrum einiger Substanzen. *Ann. Phys.*, 219:272–282, 1871.
[9] Stanford University. Types of Ray Tracing, Dec 2006. [Online, accessed 05.09.2018].
[10] R. Strugalski. Optical Table Simulator. http://www.radioactivepages.com/opticaltable.aspx, Oct 2016. [Online, accessed 13.10.2020].
[11] Xennial Digital. Physics: Optics Table. https://store.steampowered.com/app/1058410/Physics_Optics_Table/, Oct 2020. [Online, accessed 07.01.2021].
[12] ZEMAX LLC. OpticStudio. https://www.zemax.com/products/opticstudio. [Online, accessed 13.10.2020].