GPU-BASED VOLUME RAY-CASTING SUPPORTING SPECULAR REFLECTION AND REFRACTION

Timo Ropinski, Klaus Hinrichs
Visualization and Computer Graphics Research Group (VisCG), University of Münster, Germany

Jens Kasten
Konrad-Zuse-Zentrum für Informationstechnik, Berlin, Germany

Keywords: Volume rendering, GPU-based ray-casting, Advanced illumination.

Abstract: Nowadays mostly local illumination models are used when rendering volumetric data. When computing global light effects, interactive frame rates are usually hard to achieve. We present an extension of GPU-based volume ray-casting, which allows to compute specular reflection and refraction effects at interactive frame rates on current commodity graphics hardware. In contrast to other techniques proposed for integrating these effects into volume rendering, our technique does not constrain the type of rendering used, i. e., it can be used with DVR as well as isosurface rendering.

1 INTRODUCTION

Much research has been conducted in the past to achieve interactive frame rates for volume rendering on consumer graphics hardware. For example, with GPU-based volume ray-casting (Roettger et al., 2003) interactive frame rates are possible while generating a high-quality rendering. Due to these performance aspects it becomes possible to integrate more sophisticated illumination models to increase the visual realism of volume rendered images (see Figure 1).

The proposed technique modifies GPU-based ray-casting by processing a ray-caster multiple times with different entry and exit points. Thus, we are able to use arbitrary ray-caster modules, potentially supporting different rendering styles, by just transforming their input points. Since the proposed implementation exploits the capabilities of current graphics hardware and achieves interactive frame rates while supporting global illumination phenomena, we support full interactivity. Thus, the transfer function can be changed interactively, and it is possible to define different materials, e. g., more reflective or glassy ones. One important aspect is our progressive refinement of the resulting rendering. By using this refinement, it becomes possible to support specular reflection as well as refraction even on older graphics hardware by still allowing interactive exploration.

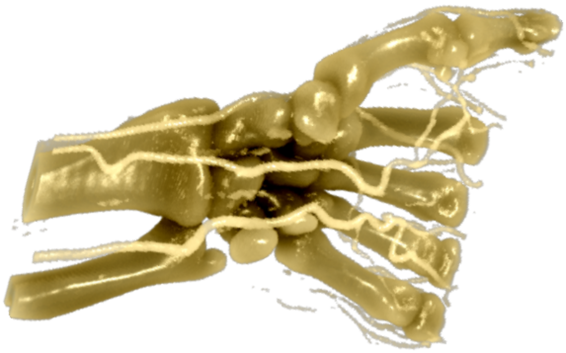
2 RELATED WORK

A lot of research has been conducted with the goal to allow interactive frame rates when ray-tracing volumetric data sets. Kajiya and von Herzen (Kajiya and Herzen, 1984) propose a volumetric ray-tracing system, which allows to simulate scattering besides the typical ray-tracing effects like specular reflection. The ray-tracing technique presented by Marmitt and Slusallek is more interactive, but constrained to isosurfaces (Marmitt and Slusallek, 2006). Since interactivity is important to be able to modify important rendering parameters, e. g., the thresholding or the transfer function, Marmitt et al. review different approaches for interactive ray-tracing of volumetric data in a state-of-the-art report (Marmitt et al., 2006).

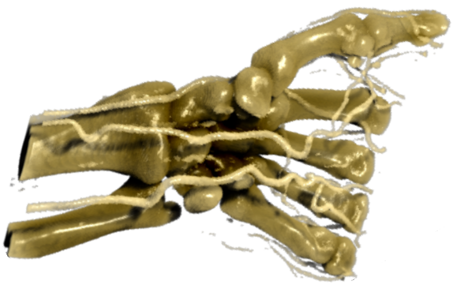
Besides ray-tracing, various publications also consider refraction in volume graphics. Rodgman and Chen describe different approaches, which exploit a ray tracer to find refractive indices of materials (Rodgman and Chen, 2001). Li and Müller aim at smooth gradients by proposing a B-spline kernel for gradient filtering (Li and Mueller, 2005).

One approach to integrate specular reflection and refraction into a GPU-based volume ray-caster has been presented by Stegmaier et al. (Stegmaier et al., 2005). They describe a ray-casting framework for generating highly appealing renderings which incor-

Ropinski T., Hinrichs K. and Kasten J. (2009).
GPU-BASED VOLUME RAY-CASTING SUPPORTING SPECULAR REFLECTION AND REFRACTION.
In *Proceedings of the Fourth International Conference on Computer Graphics Theory and Applications*, pages 219-222
DOI: 10.5220/0001784402190222



(a) specular reflections



(b) specular reflections and shadows

Figure 1: Interactive volume renderings with a higher degree of realism can be generated by exploiting specular reflection effects. A CT scan of a hand rendered with specular reflections (a). By also adding shadows the degree of realism can be further increased (b).

porate these effects. While they are more focussing on the exchangeability of different ray-casters, we aim at the combination of specular reflection and refraction effects within a single image.

3 EXTENDING GPU-BASED RAY-CASTING

3.1 Higher Order Entry and Exit Points

To integrate global illumination effects into a GPU-based volume ray-caster, we not only cast one ray per pixel, but also trace rays of higher order, i. e., we extend the ray-caster to become a ray-tracer. To be able to trace these rays, we need higher order entry and exit points for them.

The workflow of our ray traversal approach can be subdivided into four subsequent stages as shown in Figure 2. In the **first stage** the initial unmodified entry and exit points are computed as described in the previous subsection. By using these points the first order rays can be cast to generate one intermediate image as well as the computed first hit points, i. e., the positions in volume space where a ray first encounters a visible medium. The intermediate image contains the shading result achieved for each pixel, when casting a ray from the given entry point to the first hit point. While the intermediate image is cached in order to be able to blend it in the last stage, the first hit points are used as the entry points in the next recursion step. To increase the foot print of the voxels encountered at the first hit point, we alter the first hit point computation by sampling one step further into the volume. This ensures that the encountered medium is sufficiently penetrated and thus more clearly visible in the intermediate image. The exit points for the next recursion step are computed in the **second stage**, as shown in Figure 2. Based on the entry position p and the direction d of a higher order ray r, we compute the intersection between r and the bounding box of the volume. After this computation has been performed, we can hand the entry and exit points to the subsequent ray-caster, which performs the rendering in the **third stage**. When all ray-casters have finished rendering, the final image can be computed by blending all available intermediate shading images within the **fourth stage**.

To compute the ray direction d for rays reflected on a specular surface, we consider the normal of this surface. Thus, similar to the computation of the specular reflection in the Phong illumination model, the outgoing ray can be computed by considering the incoming ray and the current surface normal.

In cases where a refraction occurs, the incoming ray is bent at the surface based on Snell's law. To compute the bending angle the refraction indices of the adjacent media, for which the refraction should be computed, has to be known. When assuming that these indices are n_1 for the medium which is left by the ray, and n_2 for the medium which is entered by the ray, we can compute the bending angle θ based on the incoming angle ϕ as follows:

$$\cos(\theta) = \sqrt{1 - \left(\frac{n_1}{n_2}\right)^2 \cdot (1 - (\cos(\phi))^2)}.$$

Thus, we can compute the direction of the leaving ray A_{vec} as follows:

$$A_{vec} = \left(\frac{n_1}{n_2}\right) E_{vec} + \left(\frac{n_1}{n_2}|\cos(\phi)| - \cos(\theta)\right) N_{vec},$$

where E_{vec} is a vector representing the incoming ray and N_{vec} is the normalized gradient representing the current surface normal.

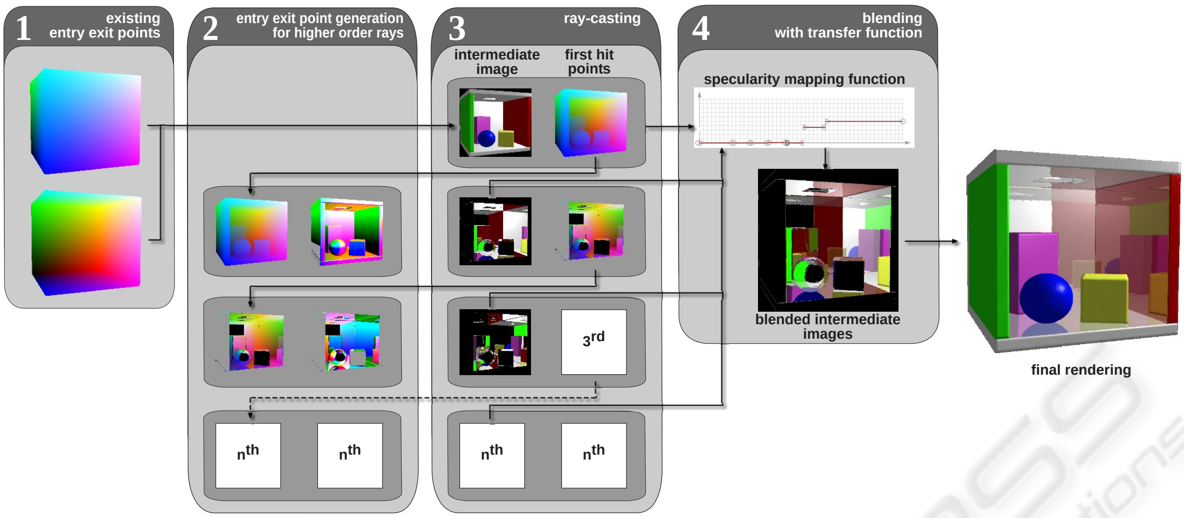


Figure 2: The workflow of the proposed technique can be subdivided into four subsequent stages. Initially the unmodified entry and exit points are computed for the first order rays (stage 1). Afterwards, for each recursive step, the exit points are calculated based on the first hit points of the previous step (stage 2). A ray-caster computes intermediate images containing the shading results for a ray segment, based on the generated entry and exit points (stage 3). Finally all intermediate images are blended into the final image using a specularity mapping function (stage 4).

In general we have to consider total reflection in cases the incoming ray hits the object under a very flat angle, i. e., no refraction but specular reflection occurs. The critical angle for which total reflection occurs can be computed as follows:

$$\phi_{crit} = \sin^{-1}\left(\frac{n_2}{n_1}\right).$$

Thus, when the incident angle ϕ exceeds ϕ_{crit}, we compute a specular reflection ray instead of a refractive one.

4 RESULTS

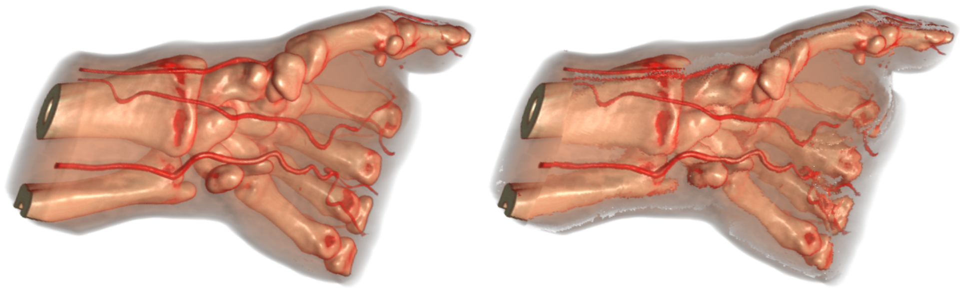
Table 1 shows the frame rates for different recursion depths and a fixed sampling rate, which is twice the maximum grid resolution of the rendered data set. To show the scalability, we have tried several data sets having different resolutions and have also altered the screen resolution. While the recursion depth has an influence on the ray length to be traversed, the latter influences the screen resolution of rays to be traced. As can be seen from the table, we still achieve interactive frame rates for moderately sized data sets when using a screen resolution of 512×512 pixels.

Parts of the hand shown in Figure 3 are rendered semi-transparently, Figure 3 (left) shows a rendering without, Figure 3 (right) with refraction. As it can be seen, refraction gives the semi-transparent medium a more glassy effect.

Table 1: The average frame rates achieved for different recursion depths with precomputed gradients, as measured on a GeForce 8800GTX graphics board. We have compared different data set sizes and screen resolutions.

data set	**recursion depth**	**screen resolution** 256^2	512^2
Hand $256 \times 256 \times 147$	0	57 fps	39 fps
	1	38 fps	22 fps
	2	30 fps	14 fps
	3	24 fps	12 fps
Teapot $256 \times 256 \times 178$	0	55 fps	38 fps
	1	40 fps	25 fps
	2	33 fps	19 fps
	3	30 fps	17 fps
Cornell Box $256 \times 256 \times 256$	0	59 fps	47 fps
	1	38 fps	19 fps
	2	31 fps	12 fps
	3	18 fps	10 fps

The proposed technique can also be combined with other advanced illumination techniques developed for volume rendering. Figure 4 shows the combination with ambient occlusion and deep shadow maps (Hadwiger et al., 2006). By exploiting these illumination techniques, a high level of realism can be achieved, while still maintaining interactive frame rates.



(a) no refraction (b) refraction

Figure 3: The hand data set rendered without refraction (a), and by applying refraction (b).

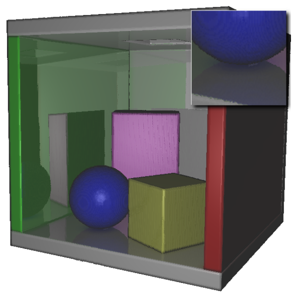


Figure 4: A synthetic Cornell box data set rendered with specular reflections, shadows and ambient occlusion.

5 CONCLUSIONS

We have proposed how to extend an interactive GPU-based volume ray-caster to support specular reflection and refraction effects. By exploiting the capabilities of current graphics hardware, our technique can be applied by still achieving interactive frame rates. The presented approach can be easily integrated into existing volume rendering frameworks, which are based on GPU-based volume ray-casting.

ACKNOWLEDGEMENTS

This work was partly supported by grants from the Deutsche Forschungsgemeinschaft (DFG), SFB 656 MoBil Münster, Germany (projects Z1, PM5). The presented concepts have been integrated into the Voreen volume rendering engine `www.voreen.org`.

REFERENCES

Hadwiger, M., Kratz, A., Sigg, C., and Bühler, K. (2006). Gpu-accelerated deep shadow maps for direct volume rendering. In *GH '06: Proceedings of the 21st ACM SIGGRAPH/Eurographics symposium on Graphics hardware*, pages 49–52, New York, NY, USA. ACM Press.

Kajiya, J. T. and Herzen, B. P. V. (1984). Ray tracing volume densities. In *SIGGRAPH '84: ACM SIGGRAPH 1984 Papers*, pages 165–174. ACM Press.

Li, S. and Mueller, K. (2005). Accelerated, high-quality refraction computations for volume graphics. In *International Workshop on Volume Graphics 2005*, pages 73–229.

Marmitt, G., Friedrich, H., and Slusallek, P. (2006). Interactive Volume Rendering with Ray Tracing. In *Eurographics State of the Art Reports*.

Marmitt, G. and Slusallek, P. (2006). Fast Ray Traversal of Tetrahedral and Hexahedral Meshes for Direct Volume Rendering. In Ertl, T., Joy, K. J., and Santos, B., editors, *Proceedings of Eurographics/IEEE-VGTC Symposium on Visualization (EuroVIS) 2006*, pages 235–242, Lisbon, Portugal. n/a.

Rodgman, D. and Chen, M. (2001). Refraction in discrete raytracing. In *International Workshop on Volume Graphics 2001*.

Roettger, S., Guthe, S., Weiskopf, D., Ertl, T., and Strasser, W. (2003). Smart hardware-accelerated volume rendering. In *VISSYM '03: Proceedings of the Symposium on Data Visualisation 2003*, pages 231–238. Eurographics Association.

Stegmaier, S., Strengert, M., Klein, T., and Ertl, T. (2005). A simple and flexible volume rendering framework for graphics-hardware–based raycasting. In *Proceedings of the International Workshop on Volume Graphics '05*, pages 187–195.