Interactive Dynamic Volume Illumination with Refraction and Caustics

Jens G. Magnus and Stefan Bruckner, *Member, IEEE Computer Society*

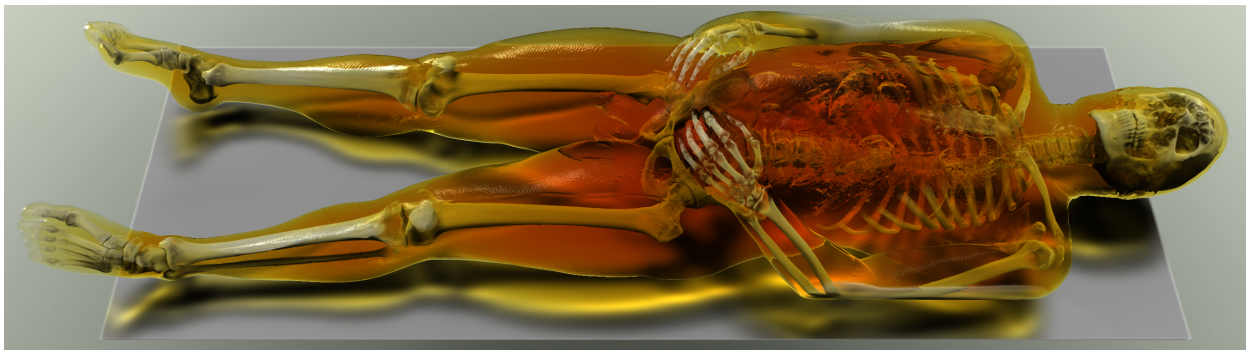

Fig. 1: The visible human CT dataset rendered using the proposed technique – the skeleton is embedded in a colored transmissive medium with higher refractive index than its surroundings.

Abstract—In recent years, significant progress has been made in developing high-quality interactive methods for realistic volume illumination. However, refraction – despite being an important aspect of light propagation in participating media – has so far only received little attention. In this paper, we present a novel approach for refractive volume illumination including caustics capable of interactive frame rates. By interleaving light and viewing ray propagation, our technique avoids memory-intensive storage of illumination information and does not require any precomputation. It is fully dynamic and all parameters such as light position and transfer function can be modified interactively without a performance penalty.

Index Terms—Interactive volume rendering, illumination, refraction, shadows, caustics

1 Introduction

Advanced illumination effects can provide important visual cues for the perception of complex spatial structures [25]. However, they are also considerably more computationally expensive than common local illumination models. This is particularly true in the context of volume data where, due to the lack of discrete homogeneous objects, illumination contributions must in principle be evaluated and propagated at every point in space. Research in recent years, partly fueled by the performance and flexibility of current GPUs, has successfully developed efficient high-quality methods for interactively rendering volume data with advanced illumination effects. However, one aspect that has been neglected so far is refraction, i.e., directional changes in light propagation due to differences in the speed of light between transmission media. Despite the fact that translucency is commonly used in the visualization of volume data, refraction effects are typically ignored and all rays are treated as straight.

Light is refracted when the speed at which it can propagate in a medium changes. When transitioning to a medium where the propagation speed is slower, the light will change direction towards the slower medium. This behavior is responsible for a wide range of optical phenomena that often significantly affect the appearance of translucent materials. For instance, the directional change towards the normal of a light ray entering a convex glass lens surrounded by air, which has a higher speed of light, is responsible for its magnifying behavior. A well-known related phenomenon are caustics, which are complex patterns of focused light surrounded by shadowed regions due to refraction by curved objects, for example on a sea floor due to a wavy ocean surface. For volumetric data, which lacks discrete object definitions, the speed of light is a continuously varying property causing a potential directional change at every point in space. Refraction effects can strongly influence the perception of transparent objects. Indeed, as studied by Fleming et al. [12], human perception is capable of reconstructing spatial properties of objects based on the distortions caused by refraction. As such, refraction may provide important additional shape cues in visualization, in particular in the context of volume visualization where nested transparent structures are commonplace.

Physically accurate rendering of scalar fields can be performed with offline methods such as volumetric photon mapping [14]. For visualization purposes, however, we require interactive performance to enable operations such as camera changes, transfer function modification, and clipping. In this paper, we present a novel approach for the interactive rendering of volume data with refractive scattering. Our approach uses a Semi-Lagrangian scheme to simultaneously propagate light and viewing rays through the volume, and hence does not require the storage of an intermediate illumination volume. Our approach supports soft shadows and caustics, as demonstrated in Figure 1, at interactive frame rates. To the best of our knowledge, our work is the first fully interactive method capable of rendering volume data using advanced refractive effects. Furthermore, our technique was designed to avoid precomputation allowing for fully dynamic manipulation of all rendering parameters.

2 Related Work

In principle, light transport in participating media is guided by the radiative transfer equation as described by Chandrasekhar [8]. However, due to the high computational costs of physically-based rendering,

• *Jens G. Magnus and Stefan Bruckner are with the University of Bergen, Norway. E-mail: jensge994@gmail.com, stefan.bruckner@uib.no.*

Manuscript received 31 Mar. 2017; accepted 1 Aug. 2017.
Date of publication 28 Aug. 2017; date of current version 1 Oct. 2017.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TVCG.2017.2744438

1077-2626 2017 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

interactive techniques typically employ simplified optical models. As discussed by Max [26], common simplifications include the emission-absorption model which disregards scattering effects or approaches that only consider single scattering. In recent years, a number of interactive volume rendering methods have been presented that incorporate more advanced effects as comprehensively discussed in the survey by Jönsson et al. [19].

Many approaches use illumination volumes to cache lighting information and recompute them only when certain properties, such as light positions or the transfer function, are changed. Ambient occlusion has been used to enhance the visualization of volumetric data due to its omnidirectional nature. To enable dynamic ambient occlusion, Ropinski et al. [33] used local data histograms to speed up the computation when the transfer function is changed. Hernell et al. [15] proposed a local variant of ambient occlusion which is only calculated in a spherical neighborhood of a voxel. Schlegel et al. [34] used summed area tables for interactive directional soft shadows. Ament et al. [3] solve full light transport within spherical regions stored in a preintegration table and combine this information with local illumination and ambient occlusion. In later work [2], they also presented a technique that uses summed area tables to compute soft shadows for multiple directional and point light sources. Sunden et al. [38] proposed a selective update scheme to minimize computations when light settings are changed. Zhang and Ma [45] presented a method to compute light transport by solving a convection-diffusion equation. They also developed an approach for decoupled shading that separates global lighting evaluation from per-sample material shading [46]. Kniss et al. [22] proposed an approach that allows for the interleaving of light and viewing computations, thereby eliminating the need for an intermediate illumination volume. Schott et al. [35] used a similar approach for a view-dependent approximation of ambient occlusion using incremental convolution. Šoltészová et al. [40] extended this approach to enable user-specified light positioning. Sunden et al. [39] proposed a plane sweep approach that also enables incremental lighting computations with a small memory footprint. Patel et al. [29] proposed an efficient convolution-based approach capable of generating dynamic soft shadows. While many efficient methods for interactive advanced volume illumination have been developed, none of the above approaches supports refraction.

Several methods for the real-time rendering of discontinuous refractive effects on surfaces have been presented. Wyman [43] used an image-space approach to approximate refraction of a distant environment through two interfaces that was also extended to handle caustics [44]. Oliveira and Brauwers [28] used a similar method to enable the rendering of refractive deformable objects. The technique by Davis and Wyman [10] additionally handles total internal reflection. The rendering of refraction effects on rough objects was investigated by Walter et al. [41], who propose a microfacet model to simulate materials such as etched glass. De Rousiers et al. [11] proposed a real-time technique capable of rendering rough refractive materials using a combination of cone tracing and macro geometry filtering together with a pre-convolved environment map.

An early approach to approximate heterogeneous distributions of refractive indices by repeated application of Snell's law was used by Berger et al. [4] to render atmospheric effects such as mirages. Gröller [13] presented a general approach to nonlinear ray tracing for the visualization of mathematical and physical systems. Weiskopf et al. [42] developed a GPU-based technique for nonlinear ray tracing in the context of relativistic visualization. Stam and Languenou [36] extended a standard ray tracing algorithm to handle non-constant media by employing the eikonal equation to model continuous variations of the refractive index in air. Full global illumination of continuously refracting participating media can be performed using volume photon mapping, as proposed by Gutierrez et al. [14]. Their approach simulates curved light paths and inelastic scattering for the rendering of atmospheric effects. Ihrke et al. [17] achieved real-time frame rates for the rendering of voxelized surface models using wavefront tracking based on the eikonal equation. Sun et al. [37] employed an octree decomposition of the refractive index field to accelerate adaptive ray marching for the relighting of refractive surface models. They exploit the sparse nature of their scenes by constructing an octree which is used to accelerate photon tracing. The illumination information is stored in a low resolution grid. The work of Hu et al. [16] presents a screen-based technique for the rendering of volumetric caustics that decomposes the scene into multiple object categories and their approach achieves high performance for scenes where relatively few pixels are affected by the caustics. Their approach also supports inhomogeneous media, but requires a very low sampling rate for interactivity. Cao et al. [7] presented an analytical solution of the ray equation of geometric optics by assuming a constant gradient of the index of refraction. Ament et al. [1] introduced the refractive radiative transfer equation which models the continuous bending of light rays for the physically-based rendering of participating media with spatially varying index of refraction. Their approach uses photon mapping to accurately simulate continuous refraction and multiple scattering. In the context of volume rendering, only few attempts have been made to incorporate refraction. Li and Mueller [24] used a spline-based filter for the high-quality reconstruction of gradients to improve the appearance of refraction effects and employed an octree to speed up computations [23], but they also focus discretely sampled surface-based objects. Rodgman and Chen [31] presented a refractive volume rendering framework based on discrete ray tracing. They use anisotropic nonlinear diffusion to filter the data in order to reduce noise in the refractive indices. Brownlee et al. [6] proposed a physically-based approach for the simulation of light refraction in flow fields. While their method demonstrates how refractive effects can be beneficial in the context of visualization, it specifically focuses on reproducing the results of experimental techniques such as schlieren and inferrometry. None of the discussed approaches, however, is capable of interactively rendering refractive volumetric scalar fields at high quality in a fully dynamic manner.

3 Background

In general, the mathematical formulation of light transport in a participating medium is given by the radiative transport equation which describes the total radiance $L(x,\omega)$ at a position x in direction ω [26]:

$$L(x,\omega) = T(x_0,x)\cdot L_0(x_0,\omega) + \int_{x_0}^{x} T(x',x)\cdot\sigma(x')\cdot L_s(x',\omega)\,dx', \quad (1)$$

where $L_0(x_0,\omega)$ is the background radiance from a boundary position x_0 and $\sigma(x)$ is the extinction coefficient at a position x. The transmittance $T(x_i,x_j)$ between any two points x_i and x_j is:

$$T(x_i,x_j) = e^{-\tau(x_i,x_j)}, \quad (2)$$

where $\tau(x_i,x_j)$ is the optical depth defined as:

$$\tau(x_i,x_j) = \int_{x_i}^{x_j} \sigma(x')\,dx'. \quad (3)$$

$L_s(x,\omega)$ corresponds to the amount of radiance arriving from all directions at a point x into a direction ω:

$$L_s(x,\omega) = \int_{\Omega} P(x,\omega',\omega)\cdot T(x_b,x)\cdot L_0(x_0,\omega')\,d\omega', \quad (4)$$

where Ω is the sphere of all directions. Scattering probability is described by the phase function $P(x,\omega',\omega)$, which is the probability density of radiance being scattered from an incident direction ω' to direction ω.

To solve the radiative transfer equation for the rendering of scalar fields, integrals are typically approximated numerically by casting rays through the volume and sampling at discrete intervals, amounting to a Riemann sum. In practice, the costly integration over the sphere is often avoided and replaced by simpler models. For instance, single scattering can be achieved by only considering the attenuated radiance from light sources or the background. Typically, this is achieved in a

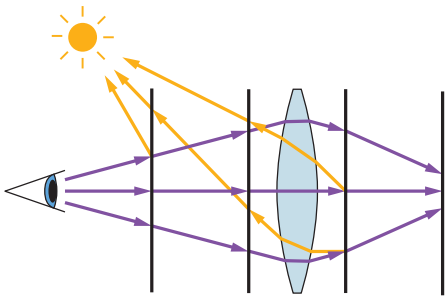

Fig. 2: Viewing rays (—) are forward integrated while light rays (—) are backward integrated. The paths of the light rays are reconstructed by looking back to the previous illumination plane.

two-pass approach where the background radiance is first attenuated by integrating the extinction along linear rays and stored in an illumination volume. In the second pass, viewing rays are cast and the illumination volume is used for shading.

When considering refraction, both light and viewing rays will not, in general, propagate along straight paths, but instead change direction based on the material properties. Specifically, refractive optics are based on Snell's law which relates the phase velocities in two media to the ratio of the sines of the angles of incidence and refraction:

$$\frac{\sin\theta_1}{\sin\theta_2} = \frac{v_1}{v_2} = \frac{\eta_2}{\eta_1}, \tag{5}$$

where θ_1 and θ_2 are the angles of incidence and refraction, respectively, v_1 and v_2 are the speed of light in the two media, and η_1 and η_2 are their indices of refraction. The index of refraction is a dimensionless material constant defined as the ratio between the speed of light in vacuum and the phase velocity of light in the medium. In standard surface ray tracing, the relationship in Equation 5 is normally used to compute the direction of the transmitted ray at an intersection point with an object. For continuously changing refractive indices, a different approach, as discussed in the next section, is better suited.

4 Interactive Volume Refraction

The goal of our method is to render refractive participating media with illumination at interactive frame rates while avoiding precomputation. In particular, we want to avoid the explicit generation of an illumination volume as, in the context of refraction, such a discretization is problematic since the curved light rays greatly complicate sampling on a regular grid. Moreover, for visualization purposes it is typically advantageous to specify the light source relative to the camera position, such that it does not have to be readjusted whenever the view is rotated. For approaches that use an illumination volume, this implies that the volume has to be recomputed whenever the camera position is changed. Similar to previous approaches [29, 40], our method therefore uses a single distant light source positioned in the same hemisphere as the viewer.

For non-interactive applications, techniques such as volumetric photon mapping [1, 14] have been employed to render scalar fields with refractive media. These techniques are, however, prohibitively expensive for interactive applications as the amount of photons needed for the result to appear continuous is too large. Rather than shooting individual photons, our technique propagates light in a plane-by-plane manner while simultaneously advancing viewing rays as illustrated in Figure 2. The volume is traversed in planes parallel to the image plane. Before discussing the details of light and viewing ray propagation in Sections 4.2, 4.3, and 4.4, we will first briefly outline the foundations of our model in Section 4.1.

4.1 Model

We assume a continuous scalar-valued volumetric function $f : \mathbb{R}^3 \to \mathbb{R}$ as well as an additional field $\eta : \mathbb{R}^3 \to \mathbb{R}$ that defines the refractive index at every point. Based on the ray equation of geometric optics [36], we can then describe the path of a light ray in this field as:

$$\frac{d}{ds}\left(\eta \frac{dx}{ds}\right) = \nabla\eta \tag{6}$$

where ds denotes an infinitesimal step in the direction tangential to the curved ray. We adopt the discretized version of this equation described by Ihrke et al. [17] and later used by Sun et al. [37]. They define the ray direction as $v = \eta \frac{dx}{ds}$ and rewrite the equation as a system of first-order differential equations:

$$\frac{dx}{ds} = \frac{v}{\eta}, \qquad \frac{dv}{ds} = \nabla\eta, \tag{7}$$

which can be discretized as:

$$x_{i+1} = x_i + \frac{\Delta s}{\eta} v_i, \qquad v_{i+1} = v_i + \Delta s \nabla\eta, \tag{8}$$

where x_i and x_{i+1} correspond to the previous and new position, respectively, along the ray, v_i and v_{i+1} are its previous and new direction, and Δs is the step size. With this we can model the changes in ray direction caused by refraction, as $\nabla\eta$ can also be discretized using finite differences. In practice, it is usually convenient to specify η as a function of the scalar value (and/or other attributes), as is commonly done for other optical properties such as color and opacity. We therefore, in addition to the color transfer function c_{tf} and the opacity transfer function α_{tf}, provide a refraction transfer function η_{tf}. Volume rendering commonly uses the particle model of Porter and Duff [30]. However, when introducing refraction it is convenient to also provide explicit control over the color of the transmissive medium. We therefore additionally introduce a medium color transfer function m_{tf}.

Following the physically-based color model by Oddy and Willis [27], we can interpret the values provided by these functions as pigment particles suspended in a filter-like medium. The color of the pigments is specified by c_{tf} while the color of the medium is given by m_{tf}. Intuitively, the value of α_{tf} controls the proportion of medium vs. particles, i.e., $\alpha_{tf} = 0$ means that no reflective particles are present, while $\alpha_{tf} = 1$ corresponds to densely packed particles that do not permit transmittance. If the medium color is constant white, this exactly corresponds to the standard Porter-Duff model. For varying medium colors, however, it implies that in this model it is possible to have a visible contribution of the volume to the final image even if α_{tf} is constant zero. An example of this would be (idealized) tinted glass which only exhibits filtering but not reflective behavior.

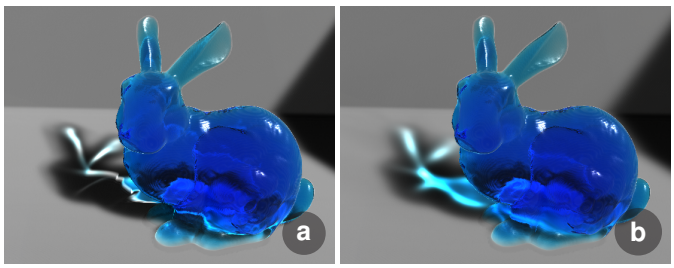

Fig. 3: Effects of light filtering. (a) No filtering. (b) Filtering of light and light direction.

4.2 Light Propagation

As discussed in the previous section, we can discretize light paths in a refracting medium by forward-integrating them using the gradient of the refractive index field. The major disadvantage of this approach is that it makes it difficult to efficiently store illumination information, which is needed during viewing ray traversal. We draw inspiration from the field of texture-based flow visualization where a similar issue occurs. When forward-advecting a texture in an unsteady vector field in a Lagrangian manner, holes may appear and a dense coverage of the

domain is difficult to maintain. A solution is to use a hybrid Lagrangian-Eulerian scheme [18], where backward integration on a regular grid is employed. Given the similarity between the two scenarios, we adopt an analogous approach for light propagation in refractive media.

Light is propagated in a plane-by-plane manner, storing its direction and radiance in 2D buffers. For every point in the light buffer, we backward-integrate along the light path by computing the intersection of a ray emanating from the current point on the light plane with the previous light plane in negative light direction, using bilinear interpolation to obtain the values for radiance and direction at the intersection point. The incoming radiance is is blended with the particle and medium contributions between the two planes, while the light direction is updated based on the gradient of the refractive index field $\nabla\eta$. More formally, for every pixel position on the current light buffer we compute:

$$L_i = L_{i-1} I_i (1-\alpha) m, \tag{9}$$

where L_i and L_{i-1} are the new and previous light color, respectively, and I_i is the light intensity (see the discussion on intensity correction below). L_0 is initialized with the color of the light source. The incoming light is attenuated by the opacity, α, and multiplied with the medium color m between the planes $i-1$ and i. We use pre-integration tables based on the corresponding transfer functions, which are updated whenever they are changed. The light direction is then updated using the ray equation of geometric optics discussed earlier:

$$ld_i = ld_{i-1} + \Delta s \nabla\eta, \tag{10}$$

where ld_i and ld_{i-1} are the new and previous light directions, respectively, and ld_0 is set to the initial light direction. The direction changes in the direction of the gradient of the field of indices of refraction $\nabla\eta$ multiplied by the distance between the light planes Δs, i.e., the integration step size.

Light filtering: This backward integration scheme as-is models a directional light source. However, following the approach of Patel et al. [29], we can use incremental convolution to efficiently support distant area light sources with controllable softness with little additional costs. The intuition of this is that by applying a blurring kernel to the previous light plane at every iteration, earlier light events, like shadowing, will become increasingly diffuse as light progresses. Their elliptical convolution kernel which uses a randomized rotational offset can be straightforwardly employed instead of a simple texture lookup, and requires only two additional texture fetches (for the three-sample kernel, which is used throughout the paper). Additionally, we use the same kernel to filter the ray directions to remedy artifacts caused by the backward mapping approach. The artifacts are caused by the fact that refracted rays tend to move towards the refractive media. The backward mapping will have a bias towards light near the boundary of the medium, where light is dispersed, because the backward ray direction will tend to point into the direction of the boundary. The filtering counteracts this bias to some degree. The filtered versions of light color and direction are then simply used instead of L_{i-1} and ld_{i-1} in Equations 9 and 10. An example of the effect of applying this filtering is shown in Figure 3.

Intensity correction: Caustics are caused by concentrations and dispersal of light and represent an important aspect of light transfer in refractive media. When used naively, the backward mapping approach lets us propagate and diffuse light, but does not capture caustics. The intensity law of geometric optics states that the energy within an infinitesimal stream tube formed from rays leaving a wavefront element of size dS_1 and hitting one of size dS_2 is constant [5]:

$$I_1 dS_1 = I_2 dS_2, \tag{11}$$

where I_1 denotes the light intensity on dS_1 and I_2 the intensity on dS_2. In our approach, we use a discretized version of the intensity law:

$$I_i = \frac{I_{i-1} S_{i-1}}{S_i}, \tag{12}$$

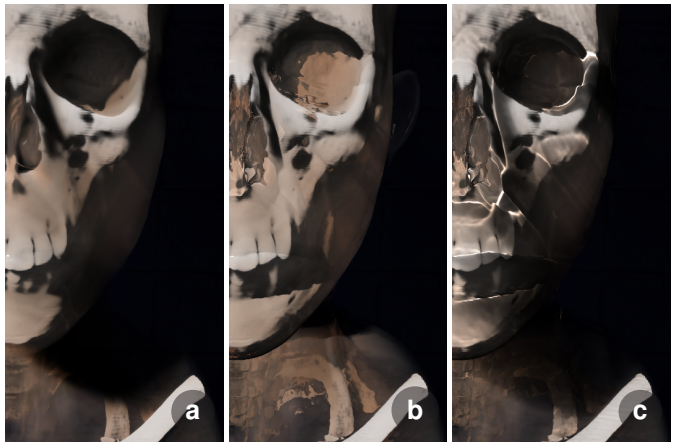

Fig. 4: Effects of light intensity correction. (a) No refraction. (b) No intensity correction. (c) Intensity correction recovers caustics.

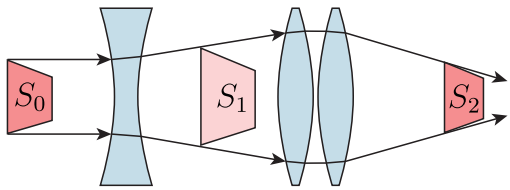

Fig. 5: Light intensity is dimmed from S_0 to S_1, and brightened again from S_1 to S_2

where I_i is then the intensity of the light on an element with an area of S_i originating from an element with an area of S_{i-1} and intensity I_{i-1}. Figure 5 illustrates the intensity law. Light leaving S_0 is dimmed as it travels to S_1, and is again brightened as it travels from S_1 to S_2.

In principle, we could compute the required areas by backward-integrating the vertices of a regular shape. However, a more efficient solution is to approximate them using screen-space partial derivatives of the intersection between the light ray and the current and previous line planes as modern GPUs offer built-in functions to compute these derivatives. The areas S_1 and S_2 in Equation 12 are then given by:

$$S_i \approx \left| \frac{d}{dx} p_i \right| \left| \frac{d}{dy} p_i \right|, \tag{13}$$

where p_i is the intersection point of the light ray and the corresponding plane. This correction is then used in Equation 12 to recover caustics.

Figure 4 shows the effect of light intensity correction. The shadows visible in Figure 4 (a), where no refraction is considered, partially disappear in Figure 4 (b) because the light is bent around the chin, but since no intensity correction is performed, the light intensity in the created gap remains the same. When correction is performed, as shown in Figure 4 (c), the shadow returns because the dispersal of the light causes its intensity be to lowered, and caustics are clearly visible.

4.3 Viewing Ray Propagation

In contrast to light propagation, viewing rays are advanced using regular forward integration, i.e., we store their position, direction, and accumulated color in a set of 2D buffers. To enable the distinction between reflective and filtering behavior as outlined in Section 4.1, we also need an additional intermediate buffer for the medium color. Each pixel in these buffers corresponds to one viewing ray. In order to propagate the viewing rays together with the light, we intersect the viewing rays with the current illumination plane.

Shading: The incoming diffuse illumination is accessed by looking up the value in the light buffer at the current location of the viewing ray. We also calculate a specular component analytically using a Cook-Torrance BRDF [9]. This calculation is done based on the direction of the light,

the viewing ray, and the normal. We use the gradient of the scalar field as the normal, and calculate the reflectivity of the interface based on the relative index of refraction between the current and previous illumination planes. This means that the specular reflections will be more prominent at interfaces with greater changes in the index of refraction, for instance at distinct material boundaries.

Compositing: Given the fact that we explicitly handle reflective and transmissive behavior, the compositing step differs from standard volume rendering. Substituting into the equations by Oddy and Willis [27, Sections 4.2 and 4.3], we obtain:

$$\begin{aligned} C_i &= C_{i-1} + (1 - A_{i-1}) \cdot M_{i-1} \cdot (\alpha \cdot c \cdot i_d + i_s) \\ A_i &= A_{i-1} + (1 - A_{i-1}) \cdot \alpha \\ M_i &= M_{i-1} \cdot m \end{aligned} \tag{14}$$

where C, A, and M, correspond to the previous (index $i-1$) and new (index i) values of particle color, opacity, and medium color, c, α, and m denote the contributions of the ray segment, and i_d and i_s correspond to the diffuse and specular illumination contributions. We note that in the original model by Oddy and Willis [27] the color was multiplied by the medium color M_{i-1} twice because they model the light as passing through the medium and being filtered once before getting reflected and passing through the medium again on the way back to the viewer. In our model, the light used for the compositing has already been filtered by the medium, so we disregard the second multiplication. Also, the specular lighting contribution is not multiplied by the opacity, meaning that purely transmissive parts of the volume without opacity contribution may still exhibit specular reflections, as is desired in the case of materials such as glass.

4.4 Environment Mapping

In contrast to common volume rendering models, where transparent structures are represented by reflecting matter, our approach can also generate visible contributions without the presence of opaque material along a viewing ray due to changes in the ray direction caused by refraction. The refracted rays will distort the background, making the shape of the distorting media more apparent. However, this effect will not be visible on a solid background. For this reason, after ray traversal has finished, we apply an optional environment mapping step where the final viewing ray directions are used to retrieve the background color from a texture that is then blended with the volume rendering based on medium and particle color and opacity. For simplicity, we employ a simple equiangular environment map, but other approaches such as a cube maps could be used as well. As neighboring rays that have undergone refraction may exhibit large differences in direction, mipmapping is needed to avoid noisy results.

Data:
lb – light buffer, *ldb* – light direction buffer,
cb – color buffer, *mb* – medium buffer,
vpb – viewing ray position buffer,
vdb – viewing ray direction buffer,
Δs – sample distance

```
1  initializeBuffers(lb,ldb,cb,mb,vpb,vdb)
2  foreach plane p_i ∈ planes do
3      writelayer = i mod 2
4      readlayer = 1 − writelayer
5      foreach fragment x ∈ fragments do
6          light propagation
7              ld_i = texture(ldb, x, readlayer)
8              lp_{i−1} = intersect(p_{i−1},x,-ld_i)
9              L_{i−1} = filter(lb, lp_{i−1}, readlayer)
10             ld_{i−1} = filter(ldb, lp_{i−1}, readlayer)
11             S_i = |dFdx(x)| · |dFdy(x)|
12             S_{i−1} = |dFdx(lp_{i−1})| · |dFdy(lp_{i−1})|
13             I_i = S_{i−1} / S_i
14             [α,m] = integrationTable(lp_{i−1},x)
15             L_i = L_{i−1} · I_i · (1 − α) · m
16             ld_i = ld_{i−1} + Δs∇η
17             store(lb, L_i, x, writelayer)
18             store(ldb, ld_i, x, writelayer)
19         end
20         viewing ray propagation
21             vp_i = texture(vpb, x, readlayer)
22             vd_i = texture(vdb, x, readlayer)
23             [C_{i−1},A_{i−1}] = texture(cb, x, readlayer)
24             M_{i−1} = texture(mb, x, readlayer)
25             i_d = texture(lb, x, readlayer)
26             i_s = specularBDRF(ld_i,vd_i,∇f)
27             [c,α,m] = integrationTable(vp_{i−1},vp_i)
28             C_i = C_{i−1} + (1 − A_{i−1}) · M_{i−1} · (α · c · i_d + i_s)
29             A_i = A_{i−1} + (1 − A_{i−1}) · α
30             M_i = M_{i−1} · m
31             vd_{i+1} = v_i + Δs∇η
32             vp_{i+1} = intersect(p_{i+1},vp_i,vd_{i+1})
33             store(vpb, vp_{i+1}, x, writelayer)
34             store(vdb, vd_{i+1}, x, writelayer)
35             store(cb, [C_i,A_i], x, writelayer)
36             store(mb, M_i, x, writelayer)
37         end
38     end
39 end
```

Algorithm 1: Pseudocode of our algorithm. For brevity, sampling of the volume and gradient reconstruction have been omitted.

5 Implementation

The presented method was implemented using OpenGL 4.5 and all its major steps are executed on the GPU. A detailed outline of the entire algorithm is given in Algorithm 1. We use 6 layered buffers with two layers each, which are written to and read from in a ping-pong fashion: the light buffer, the light direction buffer, the viewing ray position and direction buffers, and the accumulated color and medium buffers. These buffers are initialized using a compute shader (line 1). This initialization shader clears the color and medium buffers, calculates the viewing ray directions and initial positions based on the viewing transformation matrix and initializes the light and light direction buffers with the color and direction of the light source. Directions and positions are stored in viewing coordinates. Light and viewing ray propagation are then performed in a single shader program executed for every light plane. A geometry shader first constructs the light plane as a triangle strip. The fragment shader then carries the main workload of the algorithm and consists of the two stages described in Sections 4.2 and 4.3.

Light propagation: First, light propagation from the previous plane is performed using backward-integration (lines 7–8). The elliptical convolution kernel by Patel et al. [29] is then used to filter the incoming light color and direction (lines 9–10). Next, the light intensity correction is performed using the GPU's finite difference functions (lines 11–12). The contributions of the volume within the slab are determined using pre-integration tables for the medium and particle color and opacity and used to compute the new light color (lines 14–16). The index of refraction gradient is computed on-the-fly, by applying η_{tf} to the neighboring data values before computing the central differences. It is then used to update the light direction (line 16). Finally, the new light color and direction are stored in the corresponding buffers (lines 17–18).

Viewing ray propagation: First, the previous values for viewing ray position and direction, as well as the accumulated particle color, opacity, and medium color are retrieved from the corresponding buffers (lines 21–24). Next, the incoming illumination is retrieved from the light buffer and specular shading is computed using the on-the-fly computed volume gradient (lines 25–26). As in the light propagation stage, the contributions of the current ray segment are retrieved from the pre-integration tables and used to compute the new opacity, as well as particle and medium colors (lines 27–30). The ray direction is updated

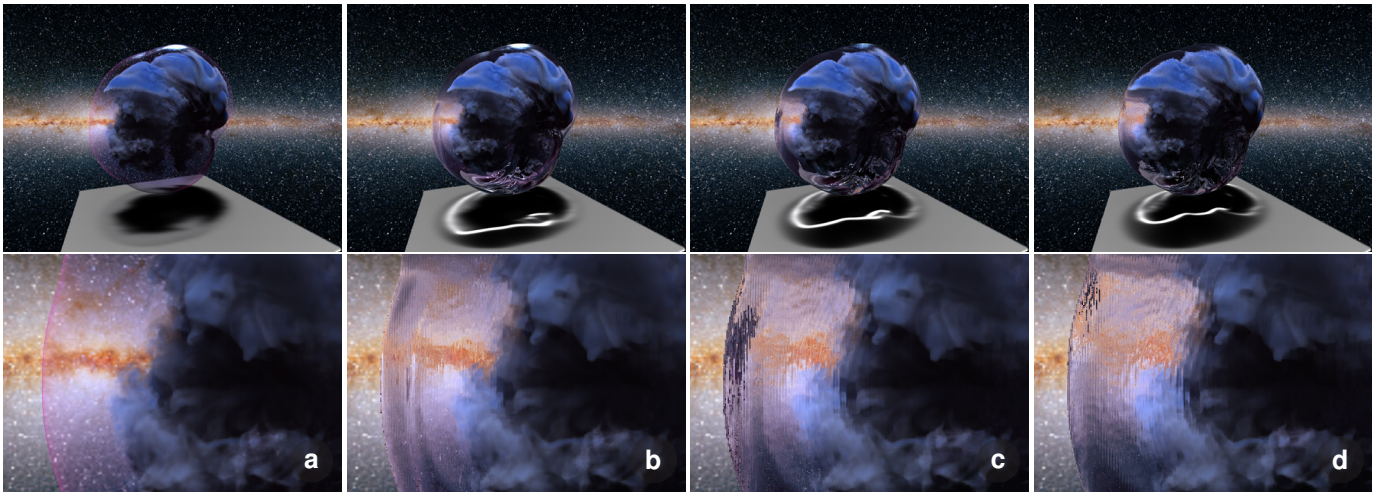

Fig. 6: One timestep of a supernova simulation with increasing refractive index of the outermost layer from (a) to (d). The bottom row shows zoom-ins of the middle left parts of the top row images.

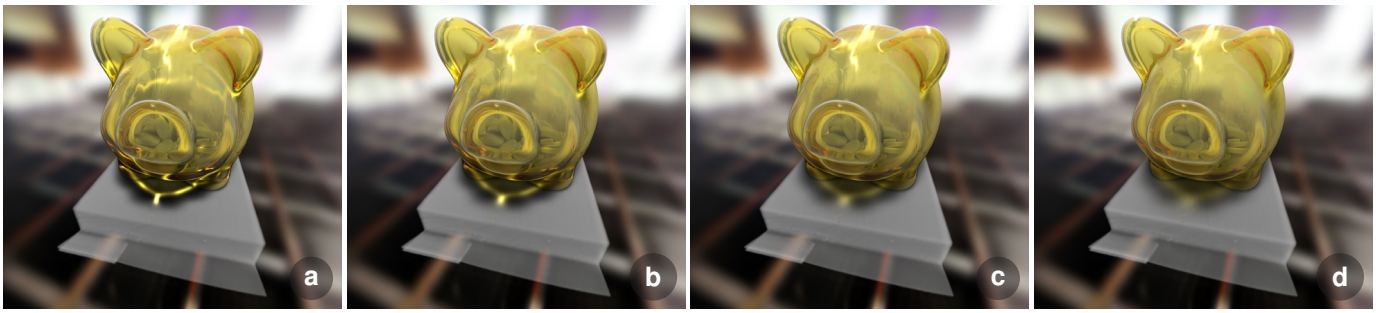

Fig. 7: CT scan of a piggy bank with refraction and combination of transmissive and reflective material properties and increasing light source softness from (a) to (d).

using the refractive index gradient (line 31) and the ray position is advanced by intersecting it with the next light plane (line 32). Finally, the new values are stored in the corresponding buffers (lines 33-36).

Our implementation uses modern OpenGL which allows for a cheap method of ping-ponging using multi-layer framebuffer attachments. By writing the gl_Layer built-in variable, a geometry shader can specify which layer of the attachment is written. Synchronization is realized with glTextureBarrier, which assures that texture writes have been completed and that texture caches are invalidated. However, other APIs such as Vulkan and Direct3D offer similar functionality and all used GPU features have been available for at least two previous hardware generations.

6 Results

In order to demonstrate the capabilities of our algorithm, we show several examples of volumetric datasets with refractive properties specified using transfer functions for particle and medium color as well as refractive index. In practice, the specification of the refractive index transfer function η_{tf} and the medium color transfer function m_{tf} is performed using an additional user interface widget, which allows for the specification of η_{tf} as a curve, while m_{tf} is specified using a color gradient. Initially, η_{tf} is constant one while m_{tf} is constant white, amounting to conventional volume rendering without refraction. While any added complexity to the already non-trivial process of transfer function specification is potentially problematic from a usability point of view, we found these functions surprisingly easy to control.

Figure 6 shows a timestep of a supernova simulation lit from above. A ground plane was added to show the effects of refractions on the shadow. The outermost layer uses a purple medium color, but is otherwise transparent revealing inner structures. In Figure 6 (a) all refractive indices are equal and the difference between the refractive indices of the outer and inner layers increases from (b) to (d). In addition to the appearance of caustics, the increasing distortion is also clearly visible in the zoom-ins on the bottom of the figure.

The piggy bank dataset, depicted in Figure 7, exhibits complex light patterns due to its curved nature. We use a combination of white reflective contributions and a yellow media color, to create the appearance of a semitransparent material such as plastic. The effect of light source softness can be seen by comparing Figure 7 (a), (b), (c), and (d) – with increasingly softer light, the caustics smoothen out and become less prominent.

Figure 8 demonstrates the importance of refraction for the appearance of transparent structures. In Figure 8 (a) a volumetric scene of a glass filled with liquid is rendered without refraction using only reflective material properties (i.e., conventional Porter-Duff compositing without a medium color). Both the glass and the liquid were given low opacity values to make them semitransparent. In Figure 8 (b), we show the same scene rendered using our method, but still with constant refraction indices, i.e., no refraction. However, we use more realistic material properties by specifying negligible absorption (i.e., opacity) for both the glass and the liquid. The colors are solely determined by the medium color. Figure 8 (c) specifies the refractive indices for both glass and liquid, but both have zero opacity simulating the appearance of a mostly transparent drink such as beer. Note the colored caustics on the ground and back planes and at the base of the stem. In Figure 8 (d), we increase the opacity of the liquid to recreate the appearance of a denser drink such as juice – in comparison to the previous figure, the liquid now absorbs most of the incoming light resulting in a dark shadow on the wall. Furthermore, a semicircular pattern caused by refraction at the rim of the glass becomes visible on the liquid. Finally,

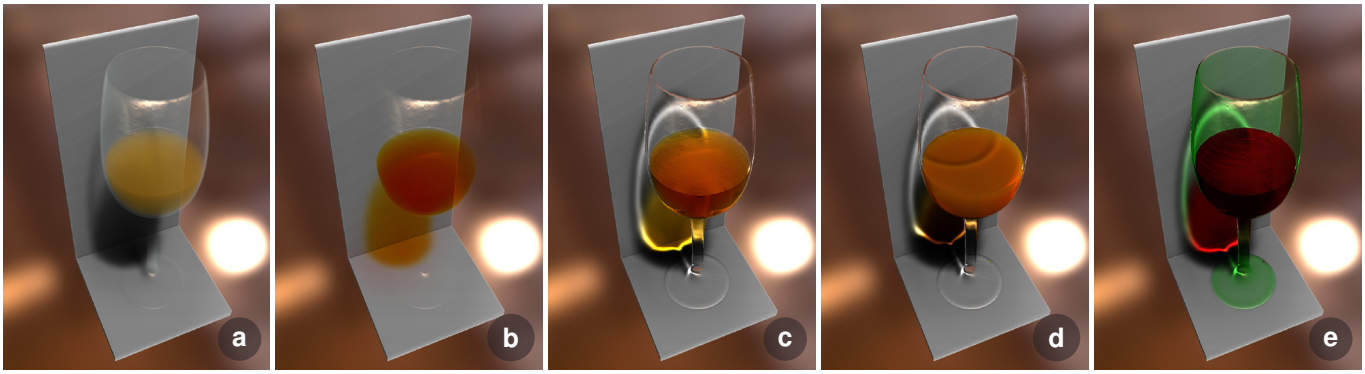

Fig. 8: A volumetric dataset of a filled glass rendered with different material properties. (a) Constant refractive index and only reflective colors, with a constant white medium color. (b) Constant refractive index with varying medium colors. (c) Different refractive indices for glass and liquid with varying medium colors (”beer”). (d) Higher absorption for the liquid (”juice”). (e) Different medium colors for glass and liquid (”red wine in green tinted glass”).

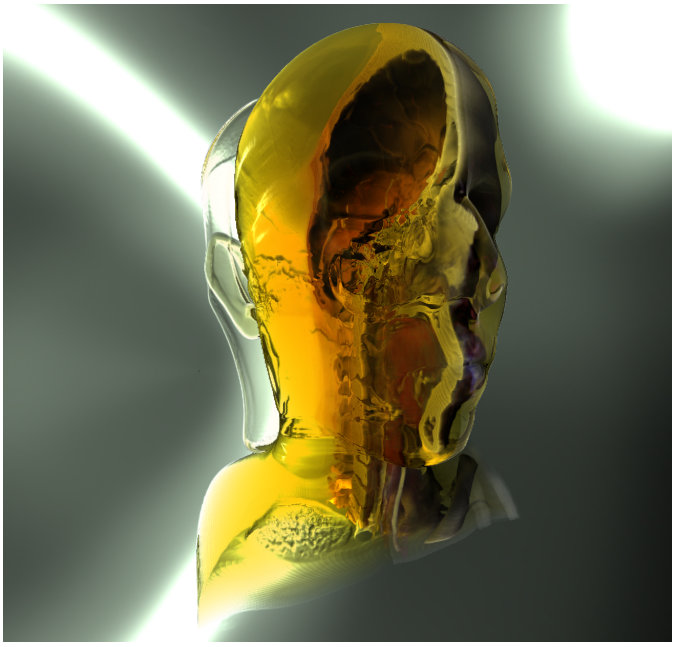

Fig. 9: CT scan of a human head with different refractive indices for air, soft tissue, and other tissues. Only the denser tissues have opacity contributions, but an additional clipping plane has been specified to set then to zero in parts of the dataset without affecting the transmissive properties.

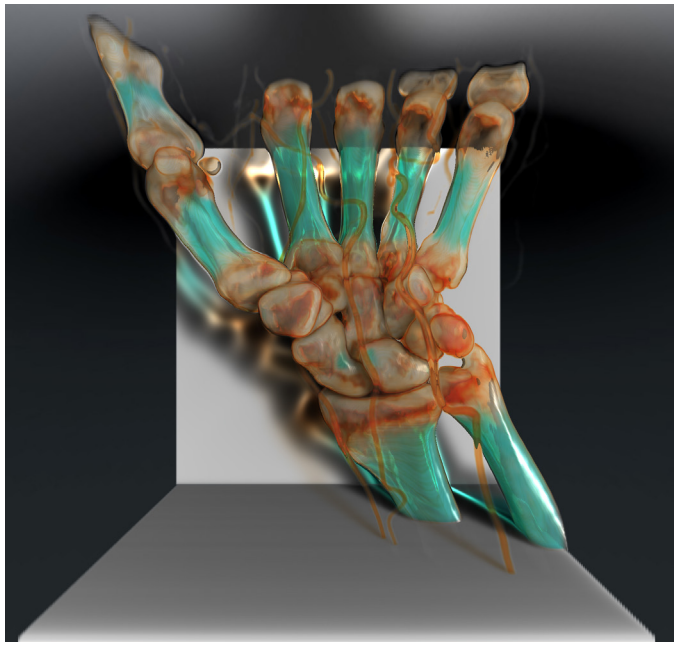

Fig. 10: CT scan of a human hand with a mixture of transmissive and reflective properties and varying refractive indices.

in Figure 8 (e) we change the medium color of the glass to a green tint and the medium color of the liquid to a red shade with similar refraction indices as in Figure 8 (c), resulting in a dark red appearance of the liquid similar to red wine in a green glass.

Figure 9 shows a CT angiography scan of a human head with different refractive indices for air and soft tissue. The medium color for the soft tissue is set to a shade of yellow, while the remaining tissues have mostly opaque reflective properties leading to an appearance similar to an amber trapping. A clipping plane has been specified to set the opacity in one half of the head to zero, but leaving all other material properties unchanged. The distortions due to refraction are clearly visible, as is the refraction on the boundary between the soft tissue and the air-filled lungs in the bottom left part of the image.

In Figure 10, a CT scan of a human hand is shown with transfer functions that use both medium and particle color to highlight different structures. The difference in refractive indices between the individual tissue types give the blood vessels the appearance of colored glass and cause a complex interplay between light and shadow regions.

In Figure 11, we show how refraction and caustics can improve the visualization of volumetric data by providing additional shape and location cues. In this example, a CT scan of a fly exhibits low contrast for fine structures such as the legs. If we increase the opacity for the corresponding data value range, undesirable occlusion of the surrounding material is introduced, as shown in Figure 11 (a). On the other hand, using a lower opacity almost completely hides these details as demonstrated in Figure 11 (b). The introduction of refraction, however, due to its non-occlusive nature, allows us to improve the situation as depicted in Figure 11 (c). Now the legs are visible due to their distortion of the background as well as the caustics caused by their curvature. In Figure 11 (d), an additional medium color is used to further improve the visibility.

Another example for the utility of refraction in visualization is the depiction of dense data. Figure 12 shows a time step from a turbulent combustion simulation. In Figure 12 (a), the chi variable is mapped to opacity, resulting in a high degree of occlusion. Lowering the opacity,

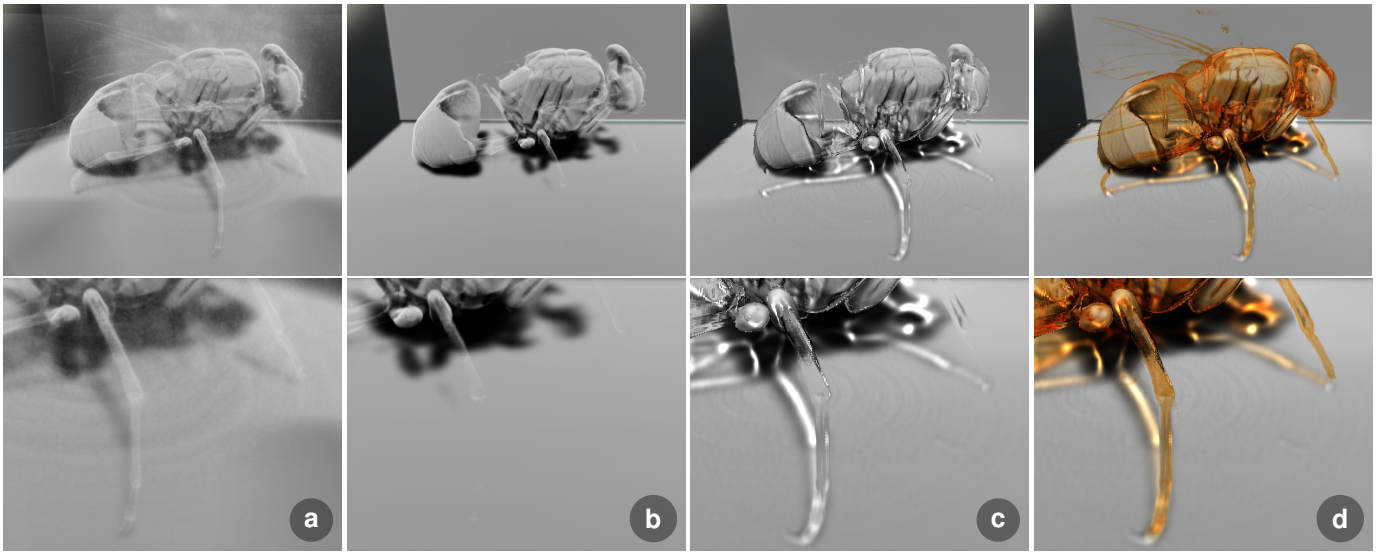

Fig. 11: A CT scan of a fly rendered without refraction in (a) and (b). Due to low contrast, it is difficult to visualize structures such as the legs without introducing occlusion. (c) Refractive material properties allow us to also depict these structures, and the resulting caustics also give a better indication of their spatial position. (d) An additional medium color further helps to clearly delineate these details.

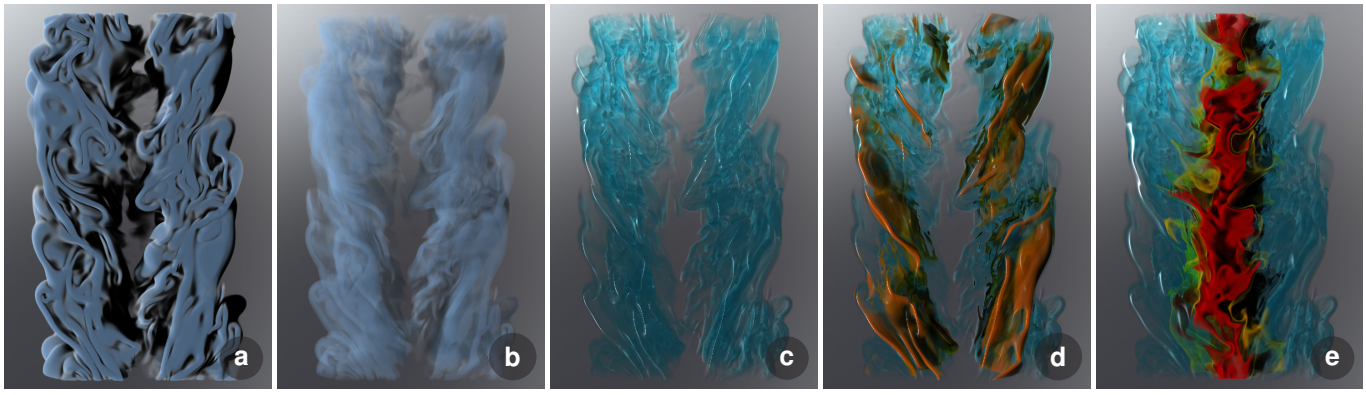

Fig. 12: A timestep of a combustion simulation. In (a) and (b), the chi variable is mapped to opacity, while in (c) the index of refraction is used instead, providing an overview visualization without introducing occlusion. (d) Reflective and refractive properties are combined to selectively highlight higher values. (e) Reflective properties are used to show the mixture fraction variable instead.

as shown in Figure 12 (b), only partially helps at the cost of reduced contrast. Using refraction, on the other hand, provides a sparse overview visualization, as depicted in Figure 12 (c). Furthermore, we can combine both approaches in a focus+context manner by only using opacity for higher data values, as shown in 12 (d). This also provides an interesting option for multivariate data, as exemplified in Figure 12 (e). Here, refraction is still used to encode the chi attribute, but opacity is instead employed to visualize the mixture fraction variable in a more prominent manner.

Finally, while it is not our goal to accurately simulate light transport in participating media, but rather to achieve plausible results at interactive frame rates, we show a comparison between our approach and a physically-based renderer in Figure 13. Figure 13 (a) shows a simple scene rendered with our technique, while Figure 13 (b) uses NVIDIA's Iray, a state-of-the-art physically-based global illumination engine. Image generation with Iray took approximately three minutes. While there are obviously a number of differences between the two images, partially caused by different material models, but also due to the fact that our renderer uses a voxelized version of the scene while the original triangle mesh was used in Iray, it can be seen that the main refraction characteristics including the caustics are approximated quite well.

To evaluate the performance of our method, we conducted measurements on an Intel Core i7-4939K 3.40 GHz CPU and an NVIDIA GeForce 1080 GTX GPU. In our implementation the sample distance, i.e., the distance between light planes, is specified as a multiplier for the minimum dimensions of a voxel – a sample distance of one ensures a minimum of one sample per voxel for all orientations and all results and in this paper were generated using this setting. Table 1 lists performance measurements of the algorithm on the datasets used to produces the images in the paper for different viewport sizes. Higher resolution datasets are rendered using a higher number of illumination planes and thus generally take longer to render. While naturally, the incorporation of refraction comes at a cost, our algorithm produces high-quality images at subsecond frame rates with all computations performed on-the-fly.

7 Discussion and Limitations

Our method was developed to enable advanced volume illumination with refraction and caustics while avoiding the explicit storage of an illumination volume. However, our hybrid Lagrangian-Eulerian light propagation scheme could also be used to precompute an illumination volume, similar to the approach described by Ropinski et al. [32], and hence it would also be possible to integrate it with other precomputed global illumination methods, such as the method by Zhang and Ma [46]. The use of an illumination volume can significantly increase the render-

Dataset	Figure	Dimensions	512^2 Viewport	768^2 Viewport	1024^2 Viewport	Graph
Visible Human	Figure 1	$587 \times 341 \times 1878$	43 ms (23.26 fps)	83 ms (12.05 fps)	130 ms (7.69 fps)	
Supernova	Figure 6	$432 \times 432 \times 432$	71 ms (14.08 fps)	95 ms (10.53 fps)	132 ms (7.58 fps)	
Piggy bank	Figure 7	$512 \times 512 \times 134$	68 ms (14.71 fps)	128 ms (7.81 fps)	207 ms (4.83 fps)	
Glass	Figure 8	$768 \times 768 \times 800$	95 ms (10.53 fps)	166 ms (6.02 fps)	186 ms (5.38 fps)	
Head	Figure 9	$512 \times 512 \times 333$	64 ms (15.63 fps)	110 ms (9.09 fps)	174 ms (5.75 fps)	
Hand	Figure 10	$244 \times 124 \times 257$	14 ms (71.43 fps)	26 ms (38.46 fps)	45 ms (22.22 fps)	
Fly	Figure 11	$498 \times 498 \times 226$	55 ms (18.18 fps)	109 ms (9.17 fps)	183 ms (5.46 fps)	
Combustion	Figure 12	$480 \times 720 \times 120$	17 ms (56.82 fps)	32 ms (31.25 fps)	52 ms (19.23 fps)	

Table 1: Rendering performance as measured on an Intel Core i7-4939K 3.40 GHz CPU equipped with an NVIDIA GeForce 1080 GTX GPU. The measurements are averages over 100 frames with a sample distance of one. The graphs in the rightmost column show the scaling behavior of the render times – the x-axis is the number of pixels.

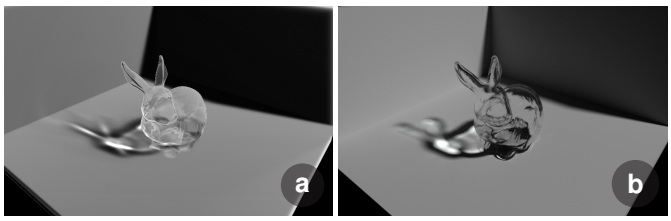

Fig. 13: Comparison of our method to a physically-based offline renderer. (a) Our technique. (b) NVIDIA's Iray.

ing performance if the light source remains static with respect to the volume. It is also possible to lower the resolution of the illumination volume for faster recomputation, however, at the cost of losing small features, as also discussed by Patel et al. [29]. For instance, features such as the fly's legs in Figure 11 could not be accurately captured with an illumination volume of 128^3 as used in the work of Sun et al. [37].

When instead performing interleaved illumination and viewing, we currently choose to orient the light plane to coincide with the image plane. However, this is not a fundamental restriction of our approach and it would also be possible to use the half angle plane between view and light direction as proposed by Kniss et al. [22]. While this would permit a 90 degree angle between light source and viewing direction, which is not possible with view-aligned traversal, it also results in more noticeable artifacts due to the fact that undersampling of the viewing rays is visually more prominent, as demonstrated by Šoltészová et al. [40]. In our experiments, the current solution has proven to be a good trade-off as artifacts only appear very close to the 90 degree angle ($>$ 85 degrees) between viewing and light direction. At present, our implementation only supports a single distant light source located in the same hemisphere as the camera. While, in principle, multiple light sources could be handled, current GPU restrictions on the number of concurrent render targets constrain the number of light buffers. Even without these constraints, however, due to the need of a single coherent traversal direction, all light sources would have to lie on the same hemisphere.

Due to the fact that lighting information has to be propagated throughout the volume, our technique currently only incorporates a limited form of empty space skipping that stops ray traversal as soon as the bounds of the contributing volume (either through opacity or through transmission) have been reached. While viewing ray computations can be skipped as soon as full opacity has been reached, illumination has to be further propagated and, due to the inherently parallel nature of execution on the GPU, this only leads to limited performance gains. In the future, we plan to investigate more effective strategies for both empty space skipping and early ray termination using features such as warp voting.

A further limitation is that our approach treats the index of refraction as a scalar quantity, disregarding the fact that physical refraction also affects the wavelength of light. This means that our method is therefore not capable of capturing wavelength-dependent phenomena such as prisms. In principle, this could be handled by specifying separate refractive indices for the spectral samples, at the cost of making their specification more complex. Alternatively, such a spectral refractive index could potentially be derived as a function of the medium color, but we have not yet experimented with this.

For some visualization applications, the effects of refraction may be undesirable. Refracting structures act as lenses, distorting the appearance of other objects. If accurate judgment of sizes or angles is important, for example in radiological applications where even perspective projection is often considered to be problematic, such effects should be avoided. On the other hand, similar to shadows, refraction effects may aid in the perception of shapes, supporting an improved communication of properties such as curvature. Studies from the perception literature show that refractive effects can improve the perception of transparent structures, even in the absence of other cues [20, 21]. Just as conventional shadows can help to judge distances, caustics can additionally assist in the judgment of curvature. One interesting aspect of refraction, in particular when combined with control over the medium color, is also that it can provide information about the presence of objects without introducing occlusion. Hence, a limited degree of refraction could represent an interesting approach for sparsely encoding the presence of contextual structures. While a detailed study of the perceptual implications of refraction effects is beyond the scope of this paper, we believe that this could be an interesting subject for further empirical research expanding on the work of Lindemann and Ropinski [25]. In this context, we would also like to note that our method does not constrain the source of the refractive index field, meaning that instead of the original scalar field a different volume can be used providing an additional optical property that can represent information, as we demonstrated in Figure 11 (e). Moreover, with respect to visualization for presentation, as opposed to analysis or exploration, we believe that selectively used refractive effects can be a meaningful stylistic tool capable of creating compelling images that was previously missing from volume visualization.

8 Conclusion

In this paper, we have presented a novel technique for the rendering of volumetric data with refraction. By interleaving light and viewing ray propagation using a hybrid backward (for illumination) and forward (for viewing rays) integration scheme, we are able to generate fully dynamic volume illumination with soft shadows and advanced effects such as caustics. Our method does not use any precomputation and all rendering parameters can be changed interactively. We have demonstrated that our technique is capable of creating plausible approximations of complex refractive phenomena in participating media that were previously only possible in offline renderers.

Acknowledgments

The research presented in this paper was supported by the MetaVis project (#250133) funded by the Research Council of Norway. The authors would like to thank Itai Kallos for informative discussions regarding the physics of our model.

REFERENCES

[1] M. Ament, C. Bergmann, and D. Weiskopf. Refractive radiative transfer equation. *ACM Transactions on Graphics*, 33(2):17:1–17:22, 2014. doi: 10.1145/2557605

[2] M. Ament, F. Sadlo, C. Dachsbacher, and D. Weiskopf. Low-pass filtered volumetric shadows. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2437–2446, 2014. doi: 10.1109/TVCG.2014.2346333

[3] M. Ament, F. Sadlo, and D. Weiskopf. Ambient volume scattering. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2936–2945, 2013. doi: 10.1109/TVCG.2013.129

[4] M. Berger, T. Trout, and N. Levit. Ray tracing mirages. *IEEE Computer Graphics and Applications*, 10(3):36–41, 1990. doi: 10.1109/38.55151

[5] M. Born and E. Wolf. *Principles of Optics: Electromagnetic Theory of Propagation, Interference and Diffraction of Light*. Cambridge University Press, Cambridge, 1999.

[6] C. Brownlee, V. Pegoraro, S. Shankar, P. McCormick, and C. D. Hansen. Physically-based interactive flow visualization based on schlieren and interferometry experimental techniques. *IEEE Transactions on Visualization and Computer Graphics*, 17(11):1574–1586, 2011. doi: 10.1109/TVCG.2010.255

[7] C. Cao, Z. Ren, B. Guo, and K. Zhou. Interactive rendering of non-constant, refractive media using the ray equations of gradient-index optics. *Computer Graphics Forum*, 29(4):1375–1382, 2010. doi: 10.1111/j.1467-8659.2010.01733.x

[8] S. Chandrasekhar. *Radiative transfer*. Dover Publications, New York, 1960.

[9] R. L. Cook and K. E. Torrance. A reflectance model for computer graphics. *ACM Transactions on Graphics*, 1(1):7–24, 1982. doi: 10.1145/357290.357293

[10] S. T. Davis and C. Wyman. Interactive refractions with total internal reflection. In *Proceedings of Graphics Interface*, pp. 185–190, 2007. doi: 10.1145/1268517.1268548

[11] C. de Rousiers, A. Bousseau, K. Subr, N. Holzschuch, and R. Ramamoorthi. Real-time rough refraction. In *Proceedings of the Symposium on Interactive 3D Graphics and Games*, pp. 111–118, 2011. doi: 10.1145/1944745.1944764

[12] R. W. Fleming, F. Jäkel, and L. T. Maloney. Visual perception of thick transparent materials. *Psychological Science*, 22(6):812–820, 2011. doi: 10.1177/0956797611408734

[13] E. Gröller. Nonlinear ray tracing: Visualizing strange worlds. *The Visual Computer*, 11(5):263–274, 1995. doi: 10.1007/BF01901044

[14] D. Gutierrez, A. Munoz, O. Anson, and F. J. Seron. Non-linear Volume Photon Mapping. In *Proceedings of the Eurographics Symposium on Rendering*, pp. 291–300, 2005. doi: 10.2312/EGWR/EGSR05/291-300

[15] F. Hernell, P. Ljung, and A. Ynnerman. Local ambient occlusion in direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 16(4):548–559, 2010. doi: 10.1109/TVCG.2009.45

[16] W. Hu, Z. Dong, I. Ihrke, T. Grosch, G. Yuan, and H.-P. Seidel. Interactive volume caustics in single-scattering media. In *Proceedings of the Symposium on Interactive 3D Graphics and Games*.

[17] I. Ihrke, G. Ziegler, A. Tevs, C. Theobalt, M. Magnor, and H.-P. Seidel. Eikonal rendering: Efficient light transport in refractive objects. *ACM Transactions on Graphics*, 26(3):59:1–59:9, 2007. doi: 10.1145/1276377.1276451

[18] B. Jobard, G. Erlebacher, and M. Y. Hussaini. Lagrangian-eulerian advection of noise and dye textures for unsteady flow visualization. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):211–222, 2002. doi: 10.1109/TVCG.2002.1021575

[19] D. Jönsson, E. Sundén, A. Ynnerman, and T. Ropinski. A survey of volumetric illumination techniques for interactive volume rendering. *Computer Graphics Forum*, 33(1):27–51, 2014. doi: 10.1111/cgf.12252

[20] T. Kawabe and R. Kogovšek. Image deformation as a cue to material category judgment. *Scientific Reports*, 7:44274, 2017. doi: 10.1038/srep44274

[21] T. Kawabe, K. Maruya, and S. Nishida. Perceptual transparency from image deformation. *PNAS*, 112(33):E4620–E4627, 2015. doi: 10.1073/pnas.1500913112

[22] J. Kniss, S. Premoze, C. Hansen, P. Shirley, and A. McPherson. A model for volume lighting and modeling. *IEEE Transactions on Visualization and Computer Graphics*, 9(2):150–162, 2003. doi: 10.1109/TVCG.2003.1196003

[23] S. Li and K. Mueller. Accelerated, high-quality refraction computations for volume graphics. In *Proceedings of Volume Graphics*, pp. 73–81, 2005. doi: 10.1109/VG.2005.194100

[24] S. Li and K. Mueller. Spline-based gradient filters for high-quality refraction computations in discrete datasets. In *Proceedings of EuroVis*, pp. 215–222, 2005. doi: 10.2312/VisSym/EuroVis05/215-222

[25] F. Lindemann and T. Ropinski. About the influence of illumination models on image comprehension in direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):1922–1931, 2011. doi: 10.1109/TVCG.2011.161

[26] N. Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995. doi: 10.1109/2945.468400

[27] R. J. Oddy and P. J. Willis. A physically based colour model. *Computer Graphics Forum*, 10(2):121–127, 1991. doi: 10.1111/1467-8659.1020121

[28] M. M. Oliveira and M. Brauwers. Real-time refraction through deformable objects. In *Proceedings of the Symposium on Interactive 3D Graphics and Games*, pp. 89–96, 2007. doi: 10.1145/1230100.1230116

[29] D. Patel, V. Šoltészová, J. M. Nordbotten, and S. Bruckner. Instant convolution shadows for volumetric detail mapping. *ACM Transactions on Graphics*, 32(5):154:1–154:18, 2013. doi: 10.1145/2492684

[30] T. Porter and T. Duff. Compositing digital images. *ACM SIGGRAPH Computer Graphics*, 18(3):253–259, 1984. doi: 10.1145/964965.808606

[31] D. Rodgman and M. Chen. Refraction in volume graphics. *Graphical Models*, 68(5):432–450, 2006. doi: 10.1016/j.gmod.2006.07.003

[32] T. Ropinski, C. Doring, and C. Rezk-Salama. Advanced volume illumination with unconstrained light source positioning. *IEEE Computer Graphics and Applications*, 30(6):29–41, 2010.

[33] T. Ropinski, J. Meyer-Spradow, S. Diepenbrock, J. Mensmann, and K. Hinrichs. Interactive volume rendering with dynamic ambient occlusion and color bleeding. *Computer Graphics Forum*, 27(2):567–576, 2008. doi: 10.1111/j.1467-8659.2008.01154.x

[34] P. Schlegel, M. Makhinya, and R. Pajarola. Extinction-based shading and illumination in GPU volume ray-casting. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):1795–1802, 2011. doi: 10.1109/TVCG.2011.198

[35] M. Schott, V. Pegoraro, C. Hansen, K. Boulanger, and K. Bouatouch. A directional occlusion shading model for interactive direct volume rendering. *Computer Graphics Forum*, 28(3):855–862, 2009. doi: 10.1111/j.1467-8659.2009.01464.x

[36] J. Stam and E. Languénou. Ray tracing in non-constant media. In *Proceedings of the Eurographics Workshop on Rendering*, pp. 225–234, 1996. doi: 10.1007/978-3-7091-7484-5_23

[37] X. Sun, K. Zhou, E. Stollnitz, J. Shi, and B. Guo. Interactive relighting of dynamic refractive objects. *ACM Transactions on Graphics*, 27(3):35:1–35:9, 2008. doi: 10.1145/1399504.1360634

[38] E. Sundén and T. Ropinski. Efficient volume illumination with multiple light sources through selective light updates. In *Proceedings of IEEE PacificVis*, pp. 231–238, 2015. doi: 10.1109/PACIFICVIS.2015.7156382

[39] E. Sundén, A. Ynnerman, and T. Ropinski. Image plane sweep volume illumination. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2125–2134, 2011. doi: 10.1109/TVCG.2011.211

[40] V. Šoltészová, D. Patel, S. Bruckner, and I. Viola. A multidirectional occlusion shading model for direct volume rendering. *Computer Graphics Forum*, 29(3):883–891, 2010. doi: 10.1111/j.1467-8659.2009.01695.x

[41] B. Walter, S. R. Marschner, H. Li, and K. E. Torrance. Microfacet models for refraction through rough surfaces. In *Proceedings of the Eurographics Symposium on Rendering*, pp. 195–206, 2007. doi: 10.2312/EGWR/EGSR07/195-206

[42] D. Weiskopf, T. Schafhitzel, and T. Ertl. GPU-based nonlinear ray tracing. *Computer Graphics Forum*, 23(3):625–633, 2004. doi: 10.1111/j.1467-8659.2004.00794.x

[43] C. Wyman. An approximate image-space approach for interactive refraction. *ACM Transactions on Graphics*, 24(3):1050–1053, 2005. doi: 10.1145/1073204.1073310

[44] C. Wyman and S. T. Davis. Interactive image-space techniques for approximating caustics. In *Proceedings of the Symposium on Interactive 3D Graphics and Games*, pp. 153–160, 2006. doi: 10.1145/1111411.1111439

[45] Y. Zhang and K.-L. Ma. Fast global illumination for interactive volume visualization. In *Proceedings of the Symposium on Interactive 3D Graphics and Games*, pp. 55–62, 2013. doi: 10.1145/2448196.2448205

[46] Y. Zhang and K.-L. Ma. Decoupled shading for real-time heterogeneous volume illumination. *Computer Graphics Forum*, 35(3):401–410, 2016. doi: 10.1111/cgf.12916