Vision, Modeling, and Visualization (2016)
M. Hullin, M. Stamminger, and T. Weinkauf (Eds.)

Real-time Novel-view Synthesis for Volume Rendering Using a Piecewise-analytic Representation

G. Lochmann[1] B. Reinert[2] A. Buchacher[1] T. Ritschel[3]

[1]Universität Koblenz-Landau [2]MPI Informatik [3]University College London

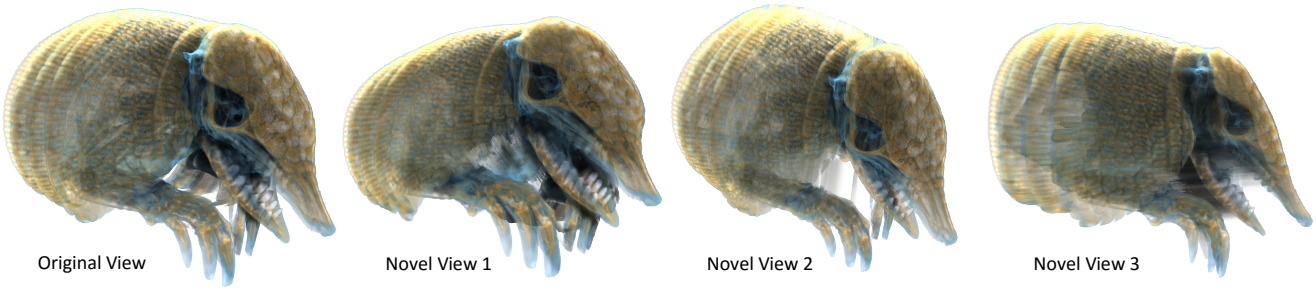

Figure 1: *Starting from an original volume rendering* (left) *our approach generates novel views in milliseconds* (right).

Abstract

Novel-view synthesis can be used to hide latency in a real-time remote rendering setup, to increase frame rate or to produce advanced visual effects such as depth-of-field or motion blur in volumes or stereo and light field imagery. Regrettably, existing real-time solutions are limited to opaque surfaces. Prior art has circumvented the challenge by making volumes opaque i. e., projecting the volume onto representative surfaces for reprojection, omitting correct volumetric effects. This paper proposes a layered image representation which is re-composed for the novel view with a special reconstruction filter. We propose a view-dependent approximation to the volume allowing to produce a typical novel view of 1024×1024 pixels in ca. 25 ms on a current GPU. At the heart of our approach is the idea to compress the complex view-dependent emission-absorption function along original view rays into a layered piecewise-analytic emission-absorption representation that can be efficiently ray-cast from a novel view. It does not assume opaque surfaces or approximate color and opacity, can be re-evaluated very efficiently, results in an image identical to the reference from the original view, has correct volumetric shading for novel views and works on a low and fixed number of layers per pixel that fits modern GPU architectures.

1. Introduction

Illuminating surfaces can be costly for complex global-illumination shaders, detailed geometry and natural lighting. Fortunately, images are often coherent across time and space: they change little between different viewpoints, and therefore, many interactive applications make use of novel-view synthesis to re-project a shaded image to a new view instead of rendering it. This allows to quickly compute latency-compensated image streams, stereo pairs or light fields and can substantially reduce the cost of depth-of-field or motion blur. Producing novel views can be done in a few milliseconds on modern graphics hardware.

Even more costly than surface shading is volume lighting. Consequently, the demand for novel-view synthesis is even more pressing. Regrettably, no such procedure to generate novel views of volume-rendered scenes exists.

In this work, we propose a technique to do so. We will extend volume rendering based on emission-absorption ray-marching to produce a custom layered image representation. This representation allows to quickly produce views different from the original one. The representation is based on a piecewise approximation of the emission-absorption from the original view. It allows for a particularly simple and fast re-projection solution, yet, with proper inter-layer and inter-view sampling. Going beyond simple view-dependent encoding of the original emission-absorption information, our approach assures that the solution of the volume rendering equation using the compact representation produces exactly the same

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

DOI: 10.2312/vmv.20161346

delivered by EUROGRAPHICS DIGITAL LIBRARY www.eg.org diglib.eg.org

image which the original representation would produce when seen from the original view. Typical volumes with many complex layers and transparent effects can be encoded into four to eight slices.

2. Related work

Volume rendering Volume rendering synthesizes images of scenes where light is not only reflected at discrete surfaces but also by particles distributed in a 3D medium [DCH88]. Such images have applications ranging from computer games over feature film production to scientific and medical visualization. Different solutions such as front-to-back and back-to-front compositing can solve the volume rendering equation (VRE) [Max95]. Using modern graphics hardware, direct volume rendering based on ray-marching is now widely available [KW03].

To accelerate volume rendering, different suggestions were made. One option is to re-use information from ray-casting the previous image to accelerate generation of new images [KSSE05]. Another popular one is to decompose the volume in a pre-process into planar layers called "Slabs" [CCF94]. Later, the layers can be quickly composed. Our approach is similar but produces the slices on-the-fly by ray-marching and storing an intermediate, non-planar volume representation that assures that both the original view can be reconstructed without error and novel views with a low error.

Another improvement often used is pre-integration [EKE01]: Instead of evaluating the integral of the transfer functions applied on the linear interpolation of two values, the result of this integration for all pairs of two values before applying the transfer function is cached. We also pre-integrate when generating the original view – be it in the presence of transfer functions or not – and use this information to improve speed and quality in a later step. Different from previous work, first, our calculation is done for every pixel at every frame, and second, the quantity integrated is converted back into emission and absorption that can be analytically integrated to produce novel views using a closed-form solution.

Recently, complex shading [HLSR08] such as indirect lighting in volumes has become a topic of interest. Full shading, e. g., using Monte Carlo [PH04, RS07] or Photon Mapping [JC98] is expensive and has complex mathematical formulations [Max95]. Our approach is basically agnostic to the global illumination solution used as long as it can be computed once for a single view point.

Re-using shading computations for new view points has been used to accelerate rendering. Deep shadow maps allow to cast shadows in scenes with transparent structures [LV00, KN01, HKSB06], precomputed radiance transfers [SKS02] can be used for volumetric effects [Rit07], radial basis functions can capture smooth indirect volume lighting [ZRL*08] and radiance caching was extended to volumes [JDZJ08]. All these approaches address how to re-use the shading, visibility or light transport term relating to κ in Sec. 3.1 in 3D, but do not solve the VRE towards the 2D pixel itself. Properly caching this result is critical to combining detailed volumes that might not even fit device memory, for low latency in novel-view synthesis and for image quality.

Encoding VRE results from the camera is similar to what deep shadow maps [LV00] do from the light's point of view [KN01, HKSB06]. However, we do not suggest a way to re-evaluate emission or absorption from the light, but from a novel view.

Novel-views Producing novel views from novel view points for 3D-rendered images was proposed by Mark et al. [MMB97]. Typically, first a buffer with one unique depth value per pixel is rendered. Two main variants exist; scattering-type [MMB97] and gathering-type computation [BMS*12]. Our approach falls into the gathering category, however, with the generalization to not only finding a unique pixel in the input that maps onto the output pixel, but to enumerate an entire distribution of colors along a view ray. Things get more complex in the presence of transparency: Typically, here, images are decomposed into layers [SGHS98], which is obvious for opaque diffuse surfaces, but more involved in the general case. For transparent reflecting or refracting surfaces solutions have been proposed [LRR*14, ZRJ*15], but these are limited to discrete surfaces at discrete depths while our paper is addressing the fully volumetric case. The fully volumetric case is the most general one. It is much sought for in remote rendering and Tomland et al. [THRS*01] have named the latency we address as a key practical challenge. When bandwidth is of no concern and the volume is static, a light field can be pre-computed from multiple view points and quickly ray-cast with depth correction to show volume data sets of opaque surfaces or multiple layers from a restricted set of views at high speed and quality [RSTK08]. Two approaches have considered producing novel views of volumes: the one of Mueller et al. [MSHC99] and the one of Zellmann et al. [ZAL12]. We compare our results to these solutions and find it to outperform them in all cases. Only if the volume rendering converges to a opaque surface, which is well-solved by classic methods [MMB97], all methods converge to a similar quality and speed.

Mueller et al. [MSHC99] use the classic slab-based volume rendering for novel-view synthesis. They suggest placing equidistant slabs [CCF94] with a coarse (16×16) height field that is later warped as a surface. Their results demonstrate applications to volumetric data but only for transfer functions that result in rather opaque surfaces. Even under this condition, substantial gaps between slabs appear (Fig. 5 in [MSHC99]), and even for solid surfaces. Our approach does not show such gaps and, using less layers, produces better images across the entire spectrum from opaque to transparent surfaces. A similar approach has been used to render on multiple devices and compensate for the latency between them [HKS10].

In the approach of Zellmann et al. [ZAL12] the volume rendering is assumed to by reducible to a single opaque surface for the purpose of re-projection. Here, every pixel has a depth and is drawn as a point, instead of using a coarse grid of quads. Zellmann et al. propose different heuristics to find the depth of these representative surfaces, such as mean, min, max of depth etc. and conclude mean depth to be a good solution for their examples. The assumption of a single depth will work for volume data sets and transfer functions that have a single dominant depth, while our solution spans the entire range between opaque and transparent media.

3. Layered Volume Re-projection

Overview Our approach has two main components (Fig. 2): The first one *synthesizes* a view-dependent volume representation of the

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

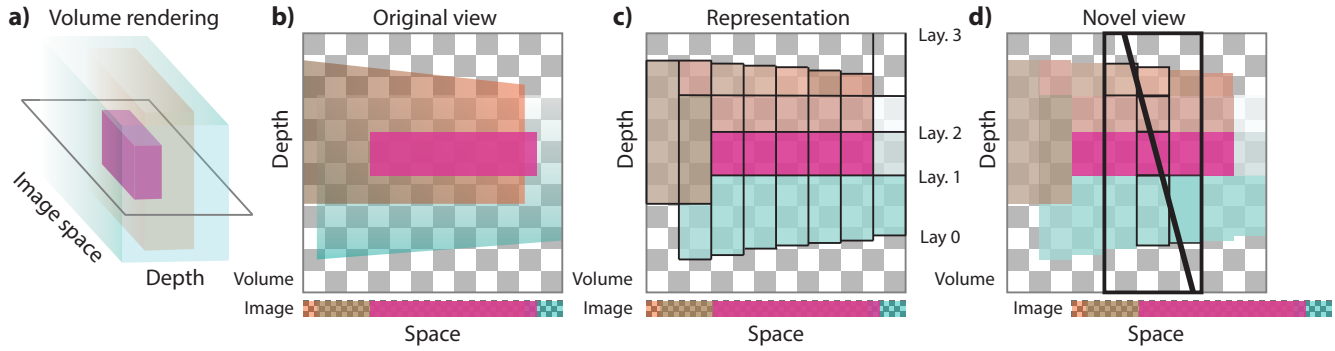

Figure 2: *Our approach from left to right in flat land. A rendering of a simple volume is shown in* (a)*. In each following part, the image space is the horizontal axis, the vertical axis depicts depth. Volume rendering reduces the two-dimensional spatial/depth-dependent emission and absorption into one-dimensional color and opacity. In the original view, our approach solves the VRE once* (b)*, including expensive user-defined shading to evaluate emission* η *and absorption* κ*. Instead of producing mere RGB output, we choose k (here,* $k = 4$*) depth layers for every pixel, shown as blocks in* (c)*. The value stored in each layer is chosen such that solving the VRE piecewise-analytic produces the same result as the reference. When producing a novel view* (d) *rays become sheared. Our approach walks the projection of the sheared ray into the original view, here three pixels. For each pixel, the solution can be computed efficiently using a piecewise analytic solution.*

original view's image (Fig. 2, c), the second one *re-synthesizes* an image for a novel view from that representation (Fig. 2, d). Both steps fit modern GPUs well. In a typical remote rendering setup, the first step occurs on the server, the second on the client.

The old-view synthesis performs one pass of volume rendering from the original view, including full shading and accessing the detailed 3D volume. It produces k depth layers for every pixel. Every pixel stores an RGBA color and a depth.

The re-synthesis uses this representation to produce an image from a novel view. It performs a heavily simplified ray-marching that makes use of the information available from the original view to produce a new one. In particular, we suggest to 2D-ray march the epipolar back-projection of the novel-view ray in the original-view.

Layering makes use of the ideas from the emission-absorption approximation we propose, so it will be described first (Sec. 3.1) and layering will be described later (Sec. 3.2). At run-time the steps are executed in the opposite order: first layers are found, then the volume inside is approximated. In any case, novel-view synthesis happens and is described last (Sec. 3.3).

3.1. Original view synthesis

The original view synthesis produces an RGBA image and a depth value for each pixel. This representation is constructed for all pixels independently and in parallel on a GPU.

For every original-view image pixel from view $\mathcal{V}_1$ exists a ray r_1. This ray is divided into n segments in a way to be defined later. Let s_1 and s_2 be the start and end distance of one such segment, respectively.

The volume rendering equation states that the radiance at a position $\mathbf{x}$ from a direction ω_o coming from the section s_1, s_2 is

$$L(\mathbf{x}, \omega_o, s_1, s_2) = \int_{s_1}^{s_2} \eta(s)\kappa(s)\mathrm{d}s,$$

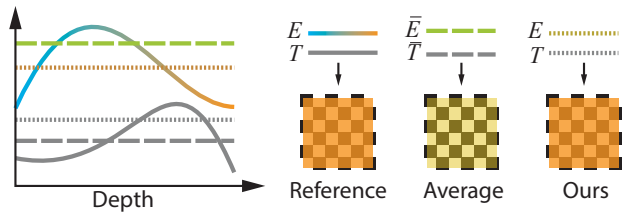

Figure 3: *Volume rendering along a single ray. The colored lines show RGB emission* $\eta(s)$*, the gray lines show extinction* $\tau(s)$*. Solving the VRE along the ray results in the reference color and opacity solution shown as patches. Simply taking the average emission and absorption* $\bar{E}$ *and* $\bar{A}$*, here shown as a dashed constant line, will result in color and opacity different from the reference. Key to our approach is choosing the constants, shown as dotted line, such that solving the VRE with them results in the reference. Still the complex functions* η *and* τ *were reduced to a single value E and A that allow for a quick analytic solution.*

the integral of the product of *emission* η and *transmittance* κ between s_1 and s_2. Transmittance between the sensor $\mathbf{x}$ and the point $\mathbf{y}(s)$ along the ray is

$$\kappa(s) = \exp(-\int_0^s \tau(\mathbf{y}(t))\mathrm{d}t$$

where τ is the *extinction* that is high if the medium is dense. Emission combines both reflected and initially emitted light and is denoted as

$$\eta(s) = L_e(\mathbf{y}(s), -\omega_o) + \int_\Omega p(\mathbf{y}(s), -\omega_o, \omega_i) L(\mathbf{y}(s), \omega_i, 0, \infty)\mathrm{d}\omega_i,$$

where L_e is initial emission, and p is the scattering function. The radiance L is associated with the RGB color C of an image pixel; the pixel opacity T (alpha value) with $\kappa(\infty)$.

The view encoding is assumed to be able to evaluate η using

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

whatever approach available to compute (global) light transport in a medium. In a client-server setting, a fast server is assumed to spend substantial computational power requiring energy and cooling on the evaluation that would not be available to the client.

The client will avoid solving the integral to save computational resources. Instead, it will make use of a simplified equation for re-projection that assumes emission (E) and absorption coefficient (A) to be constant along the ray segment. Denoting the length of the ray segment as $z = s_2 - s_1$, this results in the Beer-Lambert equations for homogeneous media

$$C = ET \qquad \text{and} \qquad T = \exp(-Az). \tag{1}$$

It is tempting to use the average emission or opacity values

$$\bar{E} = \frac{1}{z}\int_{s_1}^{s_2} \eta(\mathbf{y}(s))\mathrm{d}s \quad \text{resp.} \quad \bar{A} = \frac{1}{z}\int_{s_1}^{s_2} \tau(\mathbf{y}(s))\mathrm{d}s.$$

However, solving the VRE with average emission or opacity values $\bar{E}$ and $\bar{A}$ will not result in C and T, not even from the original view, due to the non-linearity of the VRE.

Instead, we solve for a per-ray segment emission and absorption value E and A, such that when volume-rendering a ray segment of length z from the original view, the original color C and opacity T are produced. This is achieved by re-arranging for E and A when C, T and z are given:

$$E = C/T. \qquad \text{and} \qquad A = -\log(T)/z.$$

Using these values, analytical re-rendering from the original view produces the original values, but also allows to properly render from other views, where z can be very different from the original view. This ability is key both to our quality and speed: Without a proper formulation, intervals need to be short as they do not allow extrapolation to very different views. Many short intervals are expensive to compute. Typically the intervals are much larger in depth than a pixel is wide. When seen "from the side" the values need to be corrected for, including the non-linearities.

3.2. Layering

Layering selects k depth values d_i for each ray, one for every ray segment i. This is done independently between pixels by inverting transmittance in regular steps. Formally, let $F(d) = \kappa(d)/\kappa(\infty)$ be the normalized transmittance, where the term $\kappa(\infty)$ is a normalization for pixels that do not reach full transmittance at any finite depth. We now look for values d_i that have accumulated a fraction $y_i = (i+0.5)k^{-1}$, such that $F(d_i) = y_i$. This can be achieved in two passes along the ray: The first computes the normalization $\kappa(\infty)$. The second marches the ray, updating the CDF value $F(d)$ at every location d and producing d_i at the first occasion where $F(d_i) \geq y_i$.

3.3. Novel view synthesis

Novel views are produced by efficient ray-marching on the view-dependent approximate representation devised above. Instead of making many steps on many voxels and computing emission using an expensive shading, we will now make very few steps, on very few voxels and on cached shading results. From view $\mathcal{V}_2$, again rays are cast and the color is reconstructed independently and in parallel. We will now discuss reconstructing the color for a ray r_2 through an arbitrary pixel.

First, r_2 is projected into the old view (Fig. 4, a). While there is a single pixel for every ray in the novel view, the same ray spans many pixels in the old view. Let $\mathbf{p}_{\text{start}}$ and $\mathbf{p}_{\text{end}}$ be the 2D start and end pixel locations of the epipolar line from r_2 in the old view $\mathcal{V}_1$. Starting at $\mathbf{p}_{\text{start}}$ we enumerate all pixels in the original image that intersect the 2D line.

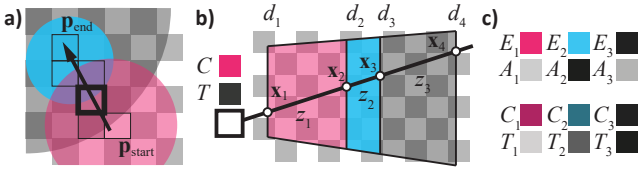

Figure 4: *Marching of a novel-view ray in a single old-view pixel in 2D.* a): *Epipolar line of one pixel in the novel view in the original view. A 2D line is rasterized and traversed using a digital differential analyzer.* b): *Processing at the one example pixel of the line. Here $k = 4$ resulting in four depths d_1 to d_4. The ray has three segments of length z_1, z_2 and z_3. The emission and absorption E and A in each layer is known* (top left)*, allowing to compute the color and opacity in each* (bottom left). c): *Front-to-back compositing produces the final color C and opacity T.*

Each pixel forms a perspective frustum in the original view (Fig. 4, b). At one pixel, we find the layer pair (d_i, d_{i+1}) in which the ray enters, and continue with (d_{i+1}, d_{i+2}), until the ray leaves the frustum again. Now, for every segment (d_i, d_{i+1}), and the novel-view ray, there is an entry and an exit position $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$. The contribution on the length $z_i = \|\mathbf{x}_i - \mathbf{x}_{i+1}\|$ can be found analytically as explained above (Eq. 1): As we know emission E and absorption A that is constant between $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$ we can compute the color C and transmittance T along the new ray in closed form. The segments are composed front-to-back (Fig. 4, c) in an inner loop and front to back along the 2D marching (not shown in Fig. 4).

The loop over the layers has either to be forward, when seeing it from the front, or backward, when seeing it from behind. The cases are separated by computing a dot product of old and new view direction.

4. Results

Here, we report qualitative (Sec. 4.1) and quantitative (Sec. 4.2) evaluation of our result, including comparisons to prior work and re-rendered references. Finally, we demonstrate several novel applications enabled by our approach (Sec. 4.3).

We refrain from reporting any speed-up over re-shading, which can be made arbitrarily favorable for any re-using approach by simply choosing sufficiently expensive shading. Our examples uses Monte Carlo volume shading from natural illumination i. e., environment maps, which requires many seconds for each image, indicting speed-up of several orders of magnitude.

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

Figure 5: *Volumetric novel-view synthesis in a game where the view from an airplane surrounded by moving clouds is controlled by a HMD. The background is sky box, while the foreground clouds are volume-lit and reprojected using our approach from a five-frame latency.*

4.1. Qualitative results

Typical results of our approach are seen in Fig. 10. We see that our approach handles both homogeneous and heterogeneous volumes, producing the right color and transparency from novel views. It allows for different types of volumes, including abstract, medical or game-like 3D volumes. Both non-dense, transparent and dense, opaque, surface-like media are possible.

Our approach produces novel views in a few milliseconds. Both speed and quality depend mainly on the difference between the old and the new view as seen in Table 2. Quality is reduced, as large changes are harder to compute, which is similar for opaque surfaces.

4.2. Quantitative results

Here we analyze the ability of our approach to predict novel views by comparing the images produced by our approach to images produced using a full re-rendering of the novel view. As test data, we use six scenes: a synthetic cube scene, four typical CT scans and a cloud as found in computer games. Images are compared using DSSIM [WBSS04] where a smaller value is better. Comparison is done for extrapolation across three different degrees of "novelty" of the novel view, measured in degree between the view directions (5°, 10° and 15°). The quality of novel views is shown in Table 2.

Additionally, we compare it to the approach of Zellmann et al. [ZAL12] and the approach of Mueller et al. [MSHC99]. While their approaches can be faster, the quality is consistently worse than ours. Additionally, we show a visual comparison of our approach with different numbers of layers in Fig. 6. We conclude, that our approach produces novel views with much higher quality at similar speed.

4.3. Applications

Novel-view synthesis has many applications. We here demonstrate for the first time novel-view synthesis-based remote, stereo and distribution-effect renderings of volumes.

Remote rendering Currently, the premiere application of novel-view synthesis is latency compensation and stereo synthesis for mobile rendering, in particular, for head-mounted displays where latency can have adverse effects on viewing comfort. In such a scenario, images are rendered on a server and sent to a mobile client. As the client constantly changes its pose, it permanently requires images that are similar, but slightly different from the transmitted pose. Our approach can compensate for this difference – and therefore the latency – by producing the view required. It can also be used to produce a stereo image pair from a monocular stream. A typical example would be a game scenario of a flight simulator involving clouds as seen in Fig. 5.

For remote rendering, the images transmitted need to be compressed in practice. The effect on the final image when applying JPEG 2000 image compression to each individual layer at a resolution of 1024×1024 and $k=2$ with 50 frames per second is seen in Fig. 7. For every layer, we need to transmit an RGB E and in mono an A and a depth value z. Instead of transmitting $E=C/T$, that can have a high dynamic range, C is transmitted and E is re-computed from T and Z on the client.

Stereo rendering A typical use of novel views is to turn one monoscopic image into a stereo image pair (Fig. 8, right). Previously, this was limited to opaque surfaces or required re-shading and re-marching the volume.

Distribution effects Distribution effects can use novel-view synthesis to convert a single image into many other images that provide samples across the lens or time domain. Averaging these results in depth-of-field (or motion blur), that is used to apply context in visualization [SPGM*11], as seen in Fig. 8, left. Previously, such images could not be produced without costly ray-marching and shading.

5. Discussion

Here we would like to discuss some important differences of our algorithm compared to previous work and potential alternatives.

Previous work In terms of previous work, our approach is able to produce novel views of volumes, while the approaches of Mueller et al. [MSHC99] and Zellmann et al. [ZAL12] were only demonstrated on volumetric data sets that effectively result in opaque surfaces. This relation is seen in Fig. 9.

Alternative solutions We have explored alternative algorithms that also use forward splatting and simpler formulations based on compositing. One such approach could just store the effective per-pixel color and transparency of every layer and compose these front-to-back from novel views. We did not consider such an approach for three reasons: First, color or transparency cannot predict well, how a ray segment would appear from a different view: If the original

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

Table 1: *Compute time (in ms, lower is better) and DSSIM error (lower is better) of our approach for different degrees of view novelty, numbers of layers* (columns) *and scenes* (rows) *seen in Fig. 1 (Zaedyus) and Fig. 10 (other). 1024×1024 on a Nvidia Geforce GTX 980.*

	5 Degree						10 Degree						15 Degree					
	2 Layer		4 Layer		8 Layer		2 Layer		4 Layer		8 Layer		2 Layer		4 Layer		8 Layer	
	Error	Speed	Error	Speed	Error	Speed	Error	Speed	Error	Speed	Error	Speed	Error	Speed	Error	Speed	Error	Speed
Cubus	.003	21.8	.002	23.4	.002	27.5	.007	23.4	.005	25.8	.003	32.9	.009	25.4	.006	29.1	.004	40.1
Zaedyus	.035	22.7	.024	24.3	.016	29.0	.049	24.5	.031	27.6	.020	35.6	.060	27.7	.037	30.9	.023	41.8
Ictiobus	.033	23.7	.018	25.6	.011	29.4	.055	24.6	.027	28.6	.015	37.4	.072	26.8	.036	33.2	.019	44.7
Celestus	.052	23.4	.024	26.7	.017	34.4	.072	26.3	.036	31.9	.023	45.3	.089	28.7	.047	37.0	.029	55.4
Cloud	.017	23.2	.010	26.7	.004	32.9	.022	25.9	.013	30.8	.007	40.8	.024	28.3	.016	35.0	.009	51.2
Walnut	.060	22.2	.041	24.1	.033	28.7	.083	24.2	.052	27.7	.039	35.5	.098	26.6	.061	31.1	.044	42.8

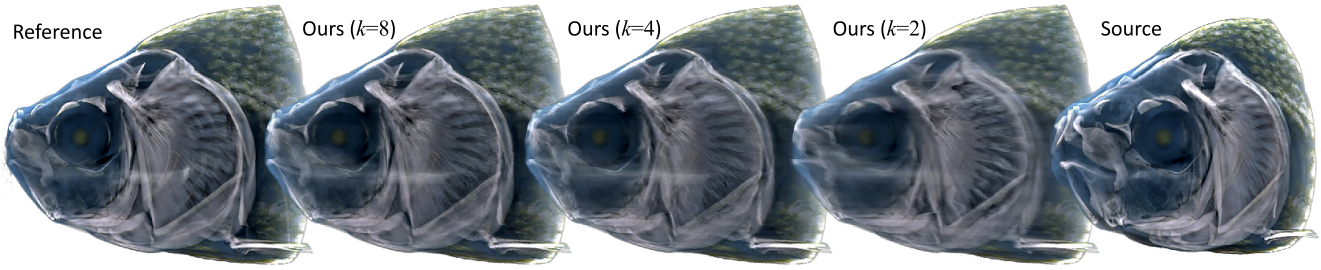

Figure 6: *A reference, results of our approach at different numbers of layers $k = \{8,4,2\}$ and the source image.*

Table 2: *Time for producing a ten-degree novel view (in ms, lower is better) and DSSIM error (lower is better) of our ($k = 4$) and other approaches (columns) for different scenes (rows).*

	Ours		[ZAL12]		[MSHC99]	
	Error	Speed	Error	Speed	Error	Speed
Cubus	.005	25.8 ms	.010	3.0 ms	.006	36.8 ms
Zaedyus	.031	27.6 ms	.081	3.0 ms	.041	33.5 ms
Ictiobus	.027	28.6 ms	.070	3.0 ms	.022	33.6 ms
Celestus	.036	31.9 ms	.067	3.0 ms	.070	40.0 ms
Cloud	.013	30.8 ms	.064	3.0 ms	.036	36.4 ms
Walnut	.052	27.7 ms	.109	2.9 ms	.063	34.4 ms

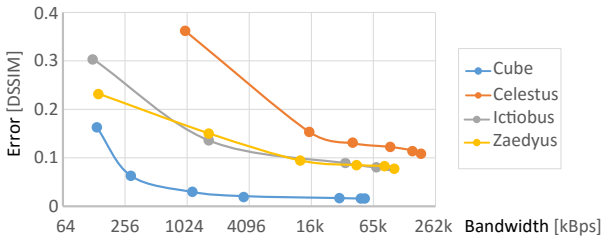

Figure 7: *The effect of image compression (x axis, log-plot) on the image quality (y axis) (DSSIM, lower is better).*

segment is very long and it appears short in the novel one it is not clear how to get the new opacity and color for the new view. Our approach can compute the correct color and opacity analytically,

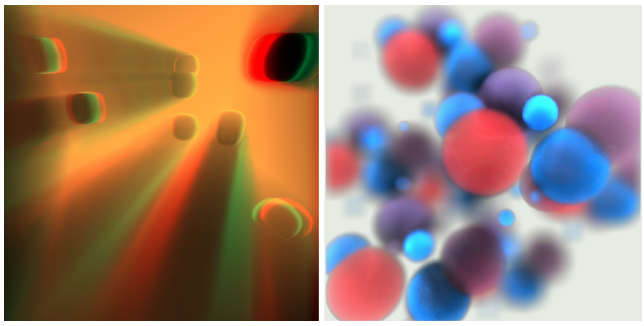

Figure 8: *Stereo image synthesis* (left) *and volumetric depth-of-field rendering* (right) *using our approach ($k = 4$).*

assuming the emission and absorption was constant. This allows for a fixed and small number of segments at each pixel. Second, composing from a novel view would require sorting the ray segments. Sorting is difficult on GPUs and typically done on the CPU, requiring CPU-GPU synchronization. Finally, our representation can be evaluated without drawing any new geometry in any specific order (splatting, or scatter-type computation). Instead, a new view is produced using the preferred gather-type computation by sampling epipolar projections of the novel ray segment in the representation.

Our layering scheme is only one out of several different alternatives. It has a few useful properties: it produces the same number of outputs, which is easily produced in parallel and later efficiently traversed. It puts details along the ray where it actually matters in terms of contribution to the original image, not just where the

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

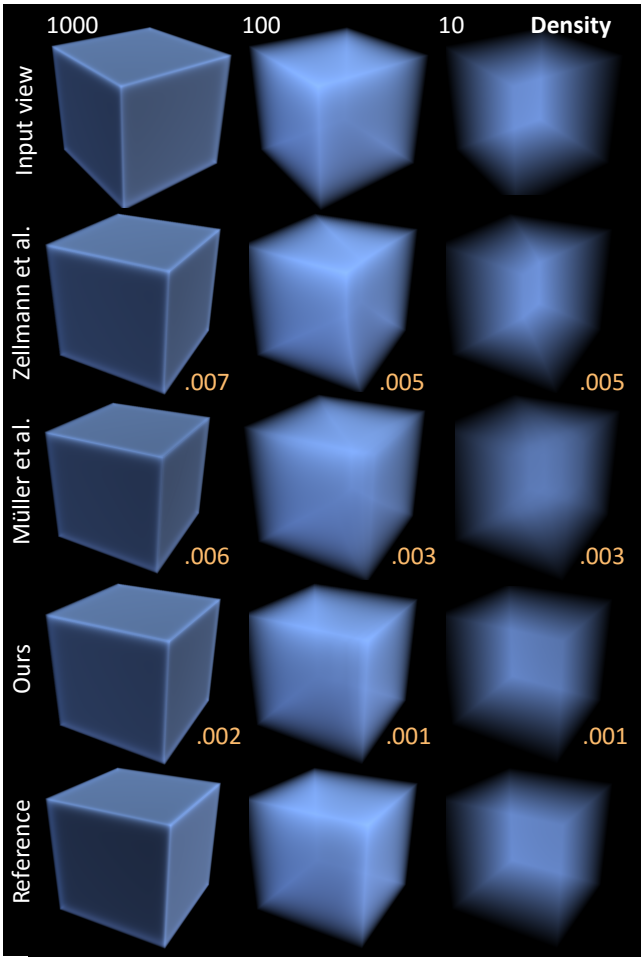

Figure 9: *Different approaches* (rows) *on different densities* (columns)*, ranging from very transparent to surface-like. Density decrease from left to right. The DSSIM (lower is better) is shown in orange. The first row is the input image; the second and third the competing approaches [ZAL12] and [MSHC99]; the fourth our approach ($k = 4$) and the fifth one a reference.*

volume has details. This is often desired, but introduces the classic surface rendering problem of occlusions into volume rendering. When a location along the ray is "behind" a dense location that has reduced transmittance to 0, it will not receive any layer. When a novel view samples this space, the information from the last contributing location will extend to infinity, which can result in "light beam"-like artifacts.

6. Conclusion

We have presented a new approach to producing images showing novel views of volumes. To this end, we adapt the volume rendering to produce multiple layers of emission and absorption, such that their re-ray-casting produces the same result, yet, it can be efficiently ray-cast also from other views to produce high-quality predictions without re-shading or access to the full 3D data set. Typical applications are remote-rendering systems, either for entertainment in computer games, but equally well for serious applications such as remote-rendering of a detailed and high quality-shaded volume data-set for a patient [THRS*01] to be inspected in-situ using a mobile device with limited bandwidth and compute power.

Our main limitation is to assume an isotropic phase function: When a medium is seen from a new view, the emission does not change. For many relevant objects, such as the silver lining on clouds this can be a visually important effect. Our approach could be extended to support such effects by representing the direction-dependent emission using Henyey-Greenstein functions such as previously done for rendering of volumetric lighting [ERWS12].

A totally different approach would be to learn how to produce novel views from example data [RRFT14]. This could be one step towards producing novel views from clouds without modifying the volume rendering code, i. e., directly from photographs.

Acknowledgements We would like to thank the Digital Morphology at the University of Texas for providing some of the data sets.

References

[BMS*12] BOWLES H., MITCHELL K., SUMNER R. W., MOORE J., GROSS M.: Iterative image warping. In *Comp. Graph. Forum (Proc. Eurographics)* (2012), vol. 31. 2

[CCF94] CABRAL B., CAM N., FORAN J.: Accelerated volume rendering and tomographic reconstruction using texture mapping hardware. In *Proc. Vis* (1994). 2

[DCH88] DREBIN R. A., CARPENTER L., HANRAHAN P.: Volume rendering. In *ACM SIGGRAPH Comp. Graph.* (1988), vol. 22. 2

[EKE01] ENGEL K., KRAUS M., ERTL T.: High-quality pre-integrated volume rendering using hardware-accelerated pixel shading. In *Proc. Graphics Harware* (2001). 2

[ERWS12] ELEK O., RITSCHEL T., WILKIE A., SEIDEL H.-P.: Interactive cloud rendering using temporally coherent photon mapping. *Computers & Graphics 36*, 8 (2012). 7

[HKS10] HAUSWIESNER S., KALKOFEN D., SCHMALSTIEG D.: Multi-frame rate volume rendering. In *EGPGV* (2010). 2

[HKSB06] HADWIGER M., KRATZ A., SIGG C., BÜHLER K.: Gpu-accelerated deep shadow maps for direct volume rendering. In *Proc. Graphics Hardware* (2006). 2

[HLSR08] HADWIGER M., LJUNG P., SALAMA C. R., ROPINSKI T.: Advanced illumination techniques for GPU volume raycasting. In *ACM SIGGRAPH Asia 2008 Courses* (2008). 2

[JC98] JENSEN H. W., CHRISTENSEN P. H.: Efficient simulation of light transport in scenes with participating media using photon maps. In *Proc. SIGGRAPH* (1998). 2

[JDZJ08] JAROSZ W., DONNER C., ZWICKER M., JENSEN H. W.: Radiance caching for participating media. *ACM Trans. Graph. 27*, 1 (2008). 2

[KN01] KIM T.-Y., NEUMANN U.: *Opacity shadow maps*. Springer, 2001. 2

[KSSE05] KLEIN T., STRENGERT M., STEGMAIER S., ERTL T.: Exploiting frame-to-frame coherence for accelerating high-quality volume raycasting on graphics hardware. In *Proc. VIS* (2005). 2

[KW03] KRUGER J., WESTERMANN R.: Acceleration techniques for gpu-based volume rendering. In *Proc IEEE Vis* (2003). 2

[LRR*14] LOCHMANN G., REINERT B., RITSCHEL T., MÜLLER S., SEIDEL H.-P.: Real-time reflective and refractive novel-view synthesis. In *Proc. VMV* (2014). 2

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.

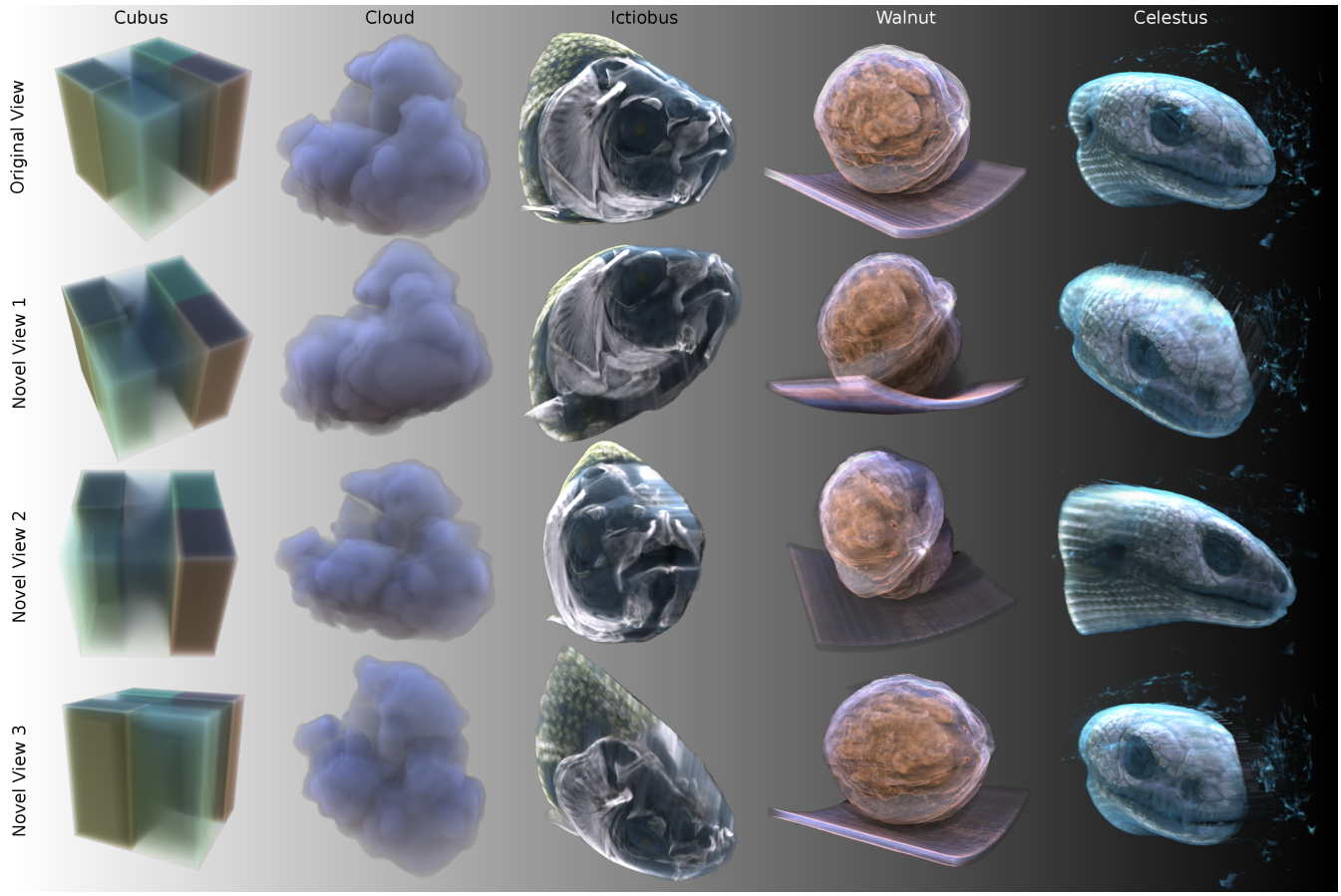

Figure 10: *Results of our approach ($k = 4$) for different views* (row two, three and four) *from an original volume rendering* (first row)

[LV00] LOKOVIC T., VEACH E.: Deep shadow maps. In *Proc. SIGGRAPH* (2000). 2

[Max95] MAX N.: Optical models for direct volume rendering. *IEEE TVCG 1*, 2 (1995). 2

[MMB97] MARK W. R., MCMILLAN L., BISHOP G.: Post-rendering 3D warping. In *Proc. ACM i3D* (1997). 2

[MSHC99] MUELLER K., SHAREEF N., HUANG J., CRAWFIS R.: Ibr-assisted volume rendering. *Proc. Vis* (1999). 2, 5, 6, 7

[PH04] PHARR M., HUMPHREYS G.: *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2004. 2

[Rit07] RITSCHEL T.: Fast GPU-based visibility computation for natural illumination of volume data sets. *Eurographics (Short Papers)* (2007). 2

[RRFT14] REMATAS K., RITSCHEL T., FRITZ M., TUYTELAARS T.: Image-based synthesis and re-synthesis of viewpoints guided by 3D models. In *CVPR* (2014). 7

[RS07] REZK-SALAMA C.: GPU-based monte-carlo volume raycasting. In *Proc. Pacific Graphics* (2007). 2

[RSTK08] REZK-SALAMA C., TODT S., KOLB A.: Raycasting of light field galleries from volumetric data. In *Comp. Graph. Forum (Proc. EuroVis)* (2008), vol. 27. 2

[SGHS98] SHADE J., GORTLER S., HE L.-W., SZELISKI R.: Layered depth images. In *Proc. SIGGRAPH* (1998). 2

[SKS02] SLOAN P.-P., KAUTZ J., SNYDER J.: Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Trans. Graph. (Proc. SIGGRAPH) 21*, 3 (2002). 2

[SPGM*11] SCHOTT M., PASCAL GROSSET A., MARTIN T., PEGORARO V., SMITH S. T., HANSEN C. D.: Depth of field effects for interactive direct volume rendering. *Comp. Graph. Forum (Proc. EuroVis) 30*, 3 (2011). 5

[THRS*01] TOMANDL B. F., HASTREITER P., REZK-SALAMA C., ENGEL K., ERTL T., HUK W. J., NARAGHI R., GANSLANDT O., NIMSKY C., EBERHARDT K. E.: Local and remote visualization techniques for interactive direct volume rendering in neuroradiology. *RadioGraphics 21*, 6 (2001). 2, 7

[WBSS04] WANG Z., BOVIK A. C., SHEIKH H. R., SIMONCELLI E. P.: Image quality assessment: from error visibility to structural similarity. *IEEE Image Processing 13*, 4 (2004). 5

[ZAL12] ZELLMANN S., AUMÜLLER M., LANG U.: Image-based remote real-time volume rendering: Decoupling rendering from view point updates. In *Proc. ASME* (2012). 2, 5, 6, 7

[ZRJ*15] ZIMMER H., ROUSSELLE F., JAKOB W., WANG O., ADLER D., JAROSZ W., SORKINE-HORNUNG O., SORKINE-HORNUNG A.: Path-space motion estimation and decomposition for robust animation filtering. *Comp. Graph. Forum (Proc. EGSR) 34*, 4 (2015). 2

[ZRL*08] ZHOU K., REN Z., LIN S., BAO H., GUO B., SHUM H.-Y.: Real-time smoke rendering using compensated ray marching. In *ACM Trans. Graph. (Proc. SIGGRAPH)* (2008), vol. 27. 2

© 2016 The Author(s)
Eurographics Proceedings © 2016 The Eurographics Association.