Volume Stylizer: Tomography-based Volume Painting

Oliver Klehm, Ivo Ihrke, Hans-Peter Seidel, Elmar Eisemann

Volume Stylizer: Tomography-based Volume Painting

Oliver Klehm* MPI Informatik
Ivo Ihrke† MPI Informatik, Saarland University
Hans-Peter Seidel‡ MPI Informatik
Elmar Eisemann§ Delft University of Technology

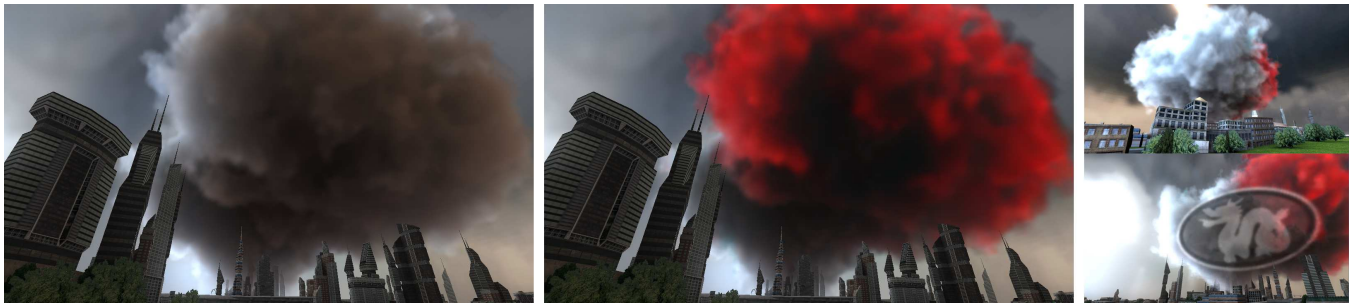

Figure 1: *Volume stylization of an environmentally lit cloud hovering over a city. On the left, we show the original cloud model, whereas on the right, the volume was stylized to increase the atmospheric tension of the scene. The modifications include the red color cast of the cloud, an increase in contrast, and the addition of a logo. Our technique optimizes for the volume properties emission and albedo from a handful of user defined images. Rendering the new volume reveals details following the user input such as the clearly visible logo.*

Abstract

Volumetric phenomena are an integral part of standard rendering, yet, no suitable tools to edit characteristic properties are available so far. Either simulation results are used directly, or modifications are high-level, e.g., noise functions to influence appearance. Intuitive artistic control is not possible.

We propose a solution to stylize single-scattering volumetric effects. Emission, scattering and extinction become amenable to artistic control while preserving a smooth and coherent appearance when changing the viewpoint. Our approach lets the user define a number of *target views* to be matched when observing the volume from this perspective. Via an analysis of the volumetric rendering equation, we can show how to link this problem to tomographic reconstruction.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

Keywords: artist control, optimization, participating media

1 Introduction

Volume rendering has long been established as a tool to enrich the appearance of virtual scenes and enabling volumetric phenomena [Kajiya and Von Herzen 1984; Max 1995; Jarosz et al. 2011; Novák et al. 2012]. Hereby, an atmospheric touch is added to otherwise "sterile" synthetic scenes. Voxels are often used to define volumes by storing physical properties such as extinction, absorption, and scattering behavior. These volumes are typically generated by means of simulation [Stam 1999] or noise functions [Perlin 1989]. While effective methods for shape control [Treuille et al. 2003; McNamara et al. 2004] exist, current solutions do not allow for modifying the appearance of such volumes, especially under complex illumination conditions.

We provide a solution that facilitates appearance control. Starting with a lit volume under environmental lighting, the user modifies (or redefines) its appearance for certain viewpoints using familiar image editing operations. Hereby, we eliminate the need on the user's side to estimate the influence of changes under complex illumination conditions. To achieve plausible results from all possible viewpoints, our approach modifies physically-based volume properties (albedo, emission, or -under special conditions- extinction) to match given appearances under a standard rendering model. While we do restrict ourselves to single-scattering, many important cues are captured and images produced with this model are convincing. Further, our solution is fast enough for interactive editing sessions with update times in the order of seconds.

Precisely, our contributions are:

- We identify properties in the radiative-transport equation and derive conditions that allow us to enable stylization via a fast linear optimization (Sec. 4.1)
- We provide an efficient implementation to keep execution time and memory cost practical (Sec. 5)
- We show several use cases to illustrate our system (Sec. 6).

2 Related Work

It is very common for artists to start with a physical simulation, tweaking it to match a desired appearance while maintaining visual plausibility. Light source editing [Schoeneman et al. 1993] or modifications to the light transport of spot lights [Kerr et al. 2010], indirect light [Obert et al. 2008], or shadows [Obert et al. 2010] have been proposed, but did not focus on participating media. Re-

*e-mail:oklehm@mpi-inf.mpg.de
†e-mail:ihrke@mpi-inf.mpg.de
‡e-mail:hpseidel@mpi-inf.mpg.de
§e-mail:e.eisemann@tudelft.nl

cently, first steps were taken to modify more complex effects like sub-surface scattering [Song et al. 2009]. While the results look impressive, the scattering is relatively strong and objects need to be rather opaque. Light beams [Nowrouzezahrai et al. 2011] also include volumetric effects, but the focus is on light modification, not on changing the properties of the medium. Further, both methods require special rendering techniques, whereas we target the application of redefining properties of a volume relevant to the radiative transport equation [Chandrasekar 1960] to ensure a plausible look for any view while keeping the rendering method unchanged.

Generating convincing volume data is known to be hard. This fact gave rise to methods that aimed at *capturing* properties of natural phenomena. These include flames [Ihrke and Magnor 2004], smoke [Hawkins et al. 2005; Ihrke and Magnor 2006] and refractive elements [Ihrke et al. 2005; Atcheson et al. 2008]. Many of them rely on tomographic reconstruction [Ihrke and Magnor 2004; Ihrke and Magnor 2006; Atcheson et al. 2008] and the ability to convincingly recover volumetric descriptions from a sparse number of views (8 to 16) hints at the possibility of defining volume properties in an image-based fashion. However, the image formation models employed in these techniques are usually simple and restricted to a single phenomenon such as emission [Ihrke and Magnor 2004; Ihrke and Magnor 2006], or refraction [Atcheson et al. 2008], only Lintu et al. [2007] attempt recovery of both, emission and absorption, for a nebula from astronomical observations. In contrast, we aim at artistic control and *modify* existing volumes to achieve a certain appearance for a low number of views under complex lighting.

This goal shares some characteristics with recent developments in *fabrication*, where objects are physically built to have a certain interaction with light in the real world. Shadows [Pauly and Mitra 2009; Baran et al. 2012], caustics [Papas et al. 2011; Papas et al. 2012], and surface reflectance [Weyrich et al. 2009] have been investigated. Others [Holroyd et al. 2011; Wetzstein et al. 2011] made it possible to show specific views when uniformly illuminating from the back and observing under particular viewing angles. The latter techniques lead to light-field display technology [Lanman et al. 2011; Wetzstein et al. 2012], or even 6D displays [Fuchs et al. 2008]. Nonetheless, these methods usually consider only a very small number of three to five layered light modulating planes, which are viewed from a well-defined viewing zone. Also, only one particular effect like absorption [Holroyd et al. 2011; Wetzstein et al. 2011] or change of polarization [Lanman et al. 2011] is considered. We enable control over appearance and achieve a full surround view under complex illumination with volumes exhibiting full resolution in all three spatial dimensions. Since we work in a rendering context, we can also use more information than for physical object manufacture and display technology. In particular, refracted ray paths are possible, where previous work is restricted to straight rays passing through a volume.

3 Background

In this section, we discuss the necessary background for our approach. In particular, we review the volumetric rendering equation and show how it, under specific conditions, can be inverted. The inversion leads to a standard tomography problem which is being solved by inverting a large scale linear system.

We start by reviewing volumetric rendering, which simulates how light interacts with participating media. We also introduce the main properties that influence the appearance. This background will be useful to understand how we offer control over appearance.

Rendering complex light transport in participating media is done by solving the radiative transport equation [Chandrasekar 1960]. It applies the physical-based cross section volume model where the

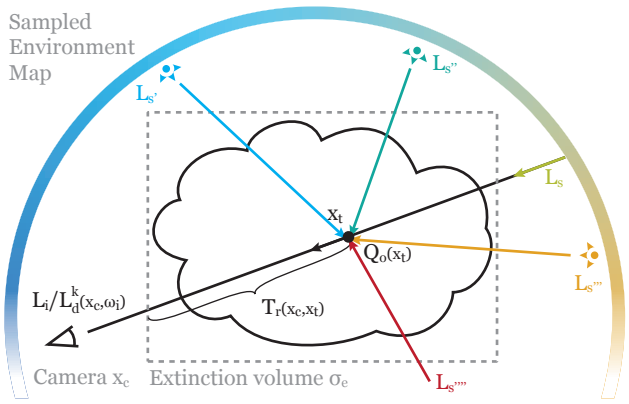

Figure 2: *Single scattering: Radiance is accumulated along the view ray. At each sample point $\mathbf{x}_t$ emitted and incoming attenuated light is scattered towards the observer ($Q_\mathrm{o} = Q_\mathrm{emit} + Q_\mathrm{scat}$).*

volume is described by infinitely small particles of random orientation. The particles can absorb, scatter, and emit light. For the moment, we do not consider refraction, which will be discussed in Sec. 5.1. We summarize all symbols in Tab. 1.

Symbol	Volume properties
σ_t	Extinction coefficient (particle per , in range $[0;\infty]$, it holds: $\sigma_\mathrm{t} = \sigma_\mathrm{s} + \sigma_\mathrm{a}$)
σ_s	Scattering coefficient (ratio of particles scattering incoming light)
σ_a	Absorption coefficient (ratio of particles absorbing incoming light)
ρ	Albedo, i. e., $\sigma_\mathrm{s}(\mathbf{x}) = \sigma_\mathrm{t}(\mathbf{x})\,\rho(\mathbf{x})$.
L_emit	Radiance emitted by the volume
f	Volumetric phase function
	Description
$\mathbf{x}, \mathbf{x}_\mathrm{s}$	Position, first surface hit by view ray or infinity
$\omega, \omega_{\mathrm{i/o}}, \Omega$	Direction, incoming/outgoing, unit sphere of directions
$L_{\mathrm{i/o}}$	Radiance incoming/outgoing
Q_o	Medium radiance outgoing from volume point
$T_r(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{b})$	$:= e^{-\int_c \sigma_\mathrm{t}(\mathbf{x}_t)dt}$ - Transmittance or probability, that a photon travels along an arc-length parameterized curve c from $\mathbf{x}_\mathrm{a}$ to $\mathbf{x}_\mathrm{b}$

Table 1: *Symbols used for volume rendering*

The light incident at a point in the scene $\mathbf{x}$ (e.g., the camera position) from direction ω_i consists of two terms:

$$L_\mathrm{i}(\mathbf{x}, \omega_\mathrm{i}) = T_r(\mathbf{x}, \mathbf{x}_\mathrm{s})\, L_\mathrm{o}(\mathbf{x}_\mathrm{s}, \omega_\mathrm{i}) + \int_c T_r(\mathbf{x}, \mathbf{x}_t)\, Q_\mathrm{o}(\mathbf{x}_t, \omega_\mathrm{i})\, dt. \quad (1)$$

The first summand is the reflected radiance at the first visible surface or emitted radiance from the background, attenuated by the volume $T_r(\mathbf{x}, \mathbf{x}_\mathrm{s})$. The second is the medium radiance $Q_\mathrm{o}(\mathbf{x}, \omega_\mathrm{o})$, which is emitted directly or out-scattered at each point $\mathbf{x}_t$ along the light path c and again attenuated by the volume. Consequently, we split $Q_\mathrm{o}(\mathbf{x}, \omega_\mathrm{o})$ into emission and scattering:

$$Q_\mathrm{o}(\mathbf{x}, \omega_\mathrm{o}) = Q_\mathrm{emit}(\mathbf{x}, \omega_\mathrm{o}) + Q_\mathrm{scat}(\mathbf{x}, \omega_\mathrm{o}). \quad (2)$$

Emission is the simpler part; each particle emits a fixed radiance independent of direction and regardless of other properties of the

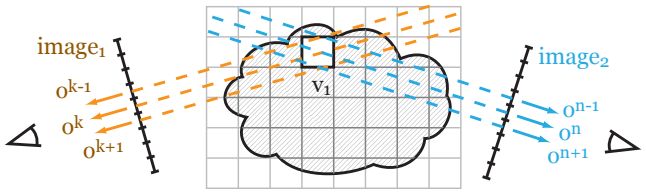

Figure 3: *Volume from two viewpoints. Problem: Changing one voxel* v_1 *influences several pixels in image*$_1$ ($\mathrm{o}^{k-1}, \mathrm{o}^k, \mathrm{o}^{k+1}$) *and in image*$_2$ ($\mathrm{o}^{n-1}, \mathrm{o}^n, \mathrm{o}^{n+1}$)

volume:

$$Q_{\mathrm{emit}}(\mathbf{x}, \omega_\mathrm{o}) = \sigma_\mathrm{t}(\mathbf{x})\, L_{\mathrm{emit}}(\mathbf{x}). \tag{3}$$

Simulating scattering requires integrating over the entire unit sphere surrounding the point to gather incoming light, which is then modulated by the phase function (the volumetric equivalent to BRDFs for surfaces; while not required by our method, we use an isotropic phase function for convenience).

$$\begin{aligned} Q_{\mathrm{scat}}(\mathbf{x}, \omega_\mathrm{o}) &= \sigma_\mathrm{s}(\mathbf{x})\, L_{\mathrm{scat}}(\mathbf{x}, \omega_\mathrm{o}) \\ &= \sigma_\mathrm{t}(\mathbf{x})\, \rho(\mathbf{x}) \int_\Omega f(\mathbf{x}, (\omega_\mathrm{i} \cdot \omega_\mathrm{o}))\, L_\mathrm{i}(\mathbf{x}, \omega_\mathrm{i})\, d\omega_\mathrm{i}. \end{aligned}$$

It is important to note that only particles that scatter light σ_s are considered. The relation between these and all particles is defined by the albedo ρ. To compute the incoming light in $L_{\mathrm{scat}}(\mathbf{x}, \omega_\mathrm{o})$, one has to apply Eq. 1 recursively, which makes volume rendering hard. For simplification, multiple volume scattering is often ignored, i. e., $L_\mathrm{i}(\mathbf{x}, \omega_\mathrm{i}) := T_r(\mathbf{x}, \mathbf{x}_\mathrm{s})\, L_\mathrm{o}(\mathbf{x}_\mathrm{s}, \omega_\mathrm{i})$; only light originating outside the volume (environmental, direct light sources…) is considered, see Fig. 2. In the following, we also apply this approximation.

4 Volume Reconstruction

To control appearance, we want to invert the volume-rendering process, upon receiving a single or multiple user-defined images defining the desired target views. We treat images as a collection of N *constraint pixels*. Given the corresponding camera view, a pixel with index $k \in [1; N]$ corresponds to a ray (origin $\mathbf{x}^k$ and direction ω^k). We denote its value $L_\mathrm{i}^k(\mathbf{x}^k, \omega^k)$ and, if applicable, the position of the first hit surface $\mathbf{x}_\mathrm{s}^k$, which otherwise is the background at ∞. We want to modify properties of a volume V, such that the *pixel constraint* is matched, namely $L_\mathrm{i}(\mathbf{x}^k, \omega^k) = L_\mathrm{i}^k(\mathbf{x}^k, \omega^k)$. For example, one could modify the emission field of V to match a given appearance.

4.1 Volume Reconstruction

A single pixel constraint can influence many voxels and, inversely, two pixel constraints might imply changes on one and the same voxel. We illustrate this situation in Fig. 3. Consequently, a perfect solution might not always be possible. Instead, we seek to find a coefficient vector $\mathbf{a} := (\dots, a^i, \dots)$ such that a linear combination $\sum_i a^i\, v^i(\mathbf{x})$ of known basis functions v^i defines a property of the volume such that the constraint pixels are matched best.

From a mathematical point of view, the basis functions could have global support. However, in practice, using a basis with spatially local support speeds up the reconstruction. One convenient choice for a basis are box functions associated to the volume's voxels, which corresponds to nearest-neighbor sampling of a 3D texture. Using triangle functions allows us to consider linearly interpolated solutions as well.

Next, we will show that the best match for emission, scattering, and extinction is the solution of a linear system.

$$\mathbf{o} = \mathbf{W}\mathbf{a}, \tag{4}$$

where $\mathbf{a}$ are the basis coefficients to be computed. The observations $\mathbf{o} := (\dots, o^k, \dots)^\mathrm{T}$ involve the constraint pixel values, and the matrix $\mathbf{W} := (\dots, \mathbf{w}^k, \dots)^\mathrm{T}$ can be derived from the volume rendering equation.

4.2 Property Reconstruction

The volume parameter reconstructions imply that Eq. 1 needs to be linearized. In the following, we derive the entries of matrix $\mathbf{W}$ necessary to estimate specific volume properties. The derivation is carried out for a single constraint pixel, i. e., for one row of matrix $\mathbf{W}$.

Emission reconstruction starts with Eq. 1 in conjunction with Eqs. 2 and 3. For a constraint pixel k, we obtain:

$$\begin{aligned} L_\mathrm{i}^k(\mathbf{x}^k, \omega^k) = T_r(\mathbf{x}^k, \mathbf{x}_\mathrm{s}^k)\, L_\mathrm{o}(\mathbf{x}_\mathrm{s}^k, \omega^k) + \\ \int_0^s T_r(\mathbf{x}^k, \mathbf{x}_t)\, (\sigma_\mathrm{t}(\mathbf{x}_t)\, L_{\mathrm{emit}}(\mathbf{x}_t) + Q_{\mathrm{scat}}(\mathbf{x}_t, \omega^k))\, dt. \end{aligned}$$

Assuming single scattering, we can split the integral into emitted and scattered light and unify all known values in o^k:

$$\begin{aligned} o^k &:= L_\mathrm{i}^k(\mathbf{x}^k, \omega^k) - T_r(\mathbf{x}^k, \mathbf{x}_\mathrm{s}^k)\, L_\mathrm{o}(\mathbf{x}_\mathrm{s}^k, \omega^k) - \\ &\qquad \int_0^s T_r(\mathbf{x}^k, \mathbf{x}_t)\, Q_{\mathrm{scat}}(\mathbf{x}_t, \omega^k)\, dt \\ &= \int_0^s T_r(\mathbf{x}^k, \mathbf{x}_t)\, \sigma_\mathrm{t}(\mathbf{x}_t)\, L_{\mathrm{emit}}(\mathbf{x}_t)\, dt. \end{aligned} \tag{5}$$

We represent $L_{\mathrm{emit}}(\mathbf{x}_t)$ by a linear combination of basis functions, whose integral can be computed:

$$\begin{aligned} o^k &= \int_0^s T_r(\mathbf{x}^k, \mathbf{x}_t)\, \sigma_\mathrm{t}(\mathbf{x}_t) \left(\sum_i L_{\mathrm{emit}}^i\, v^i(\mathbf{x}_t) \right) dt \\ &= \sum_i L_{\mathrm{emit}}^i \left(\int_0^s T_r(\mathbf{x}^k, \mathbf{x}_t)\, \sigma_\mathrm{t}(\mathbf{x}_t)\, v^i(\mathbf{x}_t)\, dt \right) \\ &=: (\vec{L}_{\mathrm{emit}} \cdot \mathbf{w}^k). \end{aligned} \tag{6}$$

The coefficients of the above equation are defined by an integral corresponding to *one* ray passing through the volume. Combining all equations defined by the constraint pixels, we obtain a linear system.

Albedo is reconstructed by establishing a similar linearization as for emission. Combining volume rendering (Eq. 1) and the constraints from a pixel k, we obtain - similar to Eq. 5:

$$\begin{aligned} o^k &:= L_\mathrm{i}^k(\mathbf{x}^k, \omega^k) - T_r(\mathbf{x}^k, \mathbf{x}_\mathrm{s}^k)\, L_\mathrm{o}(\mathbf{x}_\mathrm{s}^k, \omega^k) - \\ &\qquad \int_0^s T_r(\mathbf{x}^k, \mathbf{x}_t)\, Q_{\mathrm{emit}}(\mathbf{x}_t)\, dt \\ &= \int_0^s T_r(\mathbf{x}^k, \mathbf{x}_t)\, \sigma_\mathrm{t}(\mathbf{x}_t)\, L_{\mathrm{scat}}(\mathbf{x}_t, \omega^k)\, \rho(\mathbf{x}_t)\, dt. \end{aligned}$$

With single scattering, $L_{\mathrm{scat}}(\mathbf{x}_t, \omega^k)$ is independent of ρ, and we can solve for it. More precisely, representing the field of ρ with a

linear combination of basis functions, we obtain:

$$o^k = \sum_i \rho^i \Big(\int_0^s T_r(\mathbf{x}^k, \mathbf{x}_t)\, \sigma_\mathrm{t}(\mathbf{x}_t)\, L_\mathrm{scat}(\mathbf{x}_t, \omega^k)\, v^i(\mathbf{x}_t)\, dt \Big)$$
$$=: (\vec{\rho} \cdot \mathbf{w}^k).$$

Again, the coefficients are defined via an integral that translates to a ray marching process involving the known properties of the volume.

Emission & Albedo can be jointly optimized because L_emit and ρ are linearly independent.

Extinction can only be reconstructed if we assume that the volume's outgoing radiance is constant, i. e., $Q_\mathrm{o}(\mathbf{x}, \omega) = Q_\mathrm{o}(\mathbf{x}', \omega')\ \forall \mathbf{x}', \omega'$. As extinction is mostly used to define the overall shape of the volume and usually the first property to be derived, this restriction is usually not too problematic. Starting with Eq. 1:

$$L_\mathrm{i}^k(\mathbf{x}^k, \omega^k) = e^{-\int_0^s \sigma_\mathrm{t}(\mathbf{x}_t) dt} L_\mathrm{o}(\mathbf{x}_\mathrm{s}^k, \omega^k) + \int_0^s e^{-\int_0^t \sigma_\mathrm{t}(\mathbf{x}_{t'}) dt'}\, Q_\mathrm{o}(\mathbf{x}_t, \omega^k)\, dt.$$

We employ the linear combination of basis functions and can rewrite the first summand:

$$L_\mathrm{o}(\mathbf{x}_\mathrm{s}^k, \omega^k)\, e^{-\int_0^s \sum \sigma_\mathrm{t}^i v^i(\mathbf{x}_t) dt} = L_\mathrm{o}(\mathbf{x}_\mathrm{s}^k, \omega^k) \prod_i e^{-\sigma_\mathrm{t}^i \int_0^s v^i(\mathbf{x}_t) dt}.$$

Concerning the second summand, we exploit the constant outgoing radiance and remove Q_o from the integral. The remainder can be integrated, yielding $1 - e^{-\int_0^s \sigma_\mathrm{t}(\mathbf{x}_t) dt}$. Mathematically, the result can be shown by decomposing σ_t into a piecewise-constant approximation and splitting the outer integral accordingly [Max 1995]. Then each integral can be solved and the result recombined. This proof is valid for any Riemann-integrable extinction function. The proof also follows logically; the above remainder is the probability that a ray from the camera passes through the volume without hitting a particle, so it is one minus the probability that a ray is stopped. Now, we can use a transformation similar to the first summand to obtain:

$$L_\mathrm{i}^k(\mathbf{x}^k, \omega^k) = L_\mathrm{o}(\mathbf{x}_\mathrm{s}^k, \omega^k) \prod_i e^{-\sigma_\mathrm{t}^i \int_0^s v^i(\mathbf{x}_t) dt} + Q_\mathrm{o} \Big(1 - \prod_i e^{-\sigma_\mathrm{t}^i \int_0^s v^i(\mathbf{x}_t) dt} \Big)$$
$$= Q_\mathrm{o} + (L_\mathrm{o}(\mathbf{x}_\mathrm{s}^k, \omega^k) - Q_\mathrm{o}) \prod_i e^{-\sigma_\mathrm{t}^i \int_0^s v^i(\mathbf{x}_t) dt}.$$

Finally, we apply a logarithm to obtain a linear system in σ_t^i:

$$o^k := \ln\Big(\frac{L_\mathrm{i}^k(\mathbf{x}^k, \omega^k) - Q_\mathrm{o}}{L_\mathrm{o}(\mathbf{x}_\mathrm{s}^k, \omega^k) - Q_\mathrm{o}} \Big) = \ln\Big(\prod_i e^{-\sigma_\mathrm{t}^i \int_0^s v^i(\mathbf{x}_t) dt} \Big)$$
$$= \sum_i \ln(e^{-\sigma_\mathrm{t}^i \int_0^s v^i(\mathbf{x}_t) dt}) = \sum_i -\sigma_\mathrm{t}^i \Big(\int_0^s v^i(\mathbf{x}_t) dt \Big).$$

4.3 Discussion of Reconstruction Schemes

In all three cases that we discussed previously, we start with the volume rendering integral that contains the volume property of interest. The target field is represented as a linear combination of basis functions, which allows us to move the coefficients out of the integral, and, hereby, to isolate the unknowns. The same process is applied in computed tomography [Kak and Slaney 1988], which corresponds to our reconstruction of the extinction coefficient. However, different from medical computed tomography, we have to deal with *inconsistent* user input, i. e., for which there may be no solution that would satisfy $\mathbf{o} = \mathbf{W}\mathbf{a}$. Therefore, we opt for a solution in the least squares sense $\mathbf{W}^\mathrm{T}\mathbf{o} = \mathbf{W}^\mathrm{T}\mathbf{W}\mathbf{a}$, which means that the *optimal* solution minimizes the quadratic function $||\mathbf{W}\mathbf{a} - \mathbf{o}||^2$.

Extinction describes the overall shape of a volume, but is hard to optimize for. Emission and albedo are sufficient to change the appearance of an existing volume. Extinction cannot simultaneously be estimated in combination with emission or albedo, since it involves the solution of a complex non-linear problem. However, it is possible to begin with the reconstruction of extinction if Q_o is constant. Hence, one can solve this simplified problem in a first step. Based on this result, one can then add scattering light and refine the appearance by estimating albedo and emission while lifting the constraint that Q_o is constant.

5 Implementation

In order to ensure a quick feedback to the user, we map our optimization to the GPU via compute shaders in OpenGL. However, this mapping is not direct, as special care is needed regarding memory and multi-threading management.

Further, the matrix $\mathbf{W}$ is large (total count of constraints pixels $\times$ number of voxels), the linear system is ill-conditioned, and, finally, we seek a physically plausible, i. e., a non-negative solution.

Fortunately, $\mathbf{W}$ is sparse because each row is derived by a ray passing through the volume, intersecting only a low number of basis functions v. Still it would be too large to keep in memory. Instead, we implicitly solve the system by performing a conjugate gradient minimization of the quadratic function $||\mathbf{W}\mathbf{a} - \mathbf{o}||^2$. All necessary steps of the conjugate gradient method are carried out on the GPU and involve 3D textures to represent the vectors. Operations on these vectors are implemented as shaders.

We employ the conjugate gradient method [Shewchuk 1994] due to its fast convergence, but, for clarity, we illustrate the required operations by describing a standard gradient descent which yields the same minimum. The main ingredients are the computation of the gradient $\mathbf{W}^\mathrm{T}(\mathbf{W}\mathbf{a} - \mathbf{o})$, an update of our current solution $\mathbf{a}$ by adding a scaled version of it and iteration. Performing the update is straightforward, but the computation of the gradient is not.

To understand $\mathbf{W}^\mathrm{T}(\mathbf{W}\mathbf{a} - \mathbf{o})$, we examine its elements step by step. The matrix $\mathbf{W}$ encodes volume rendering (e.g., for Eq. (6)): $\mathbf{W}\mathbf{a} = \int_0^s T_r(\mathbf{x}^k, \mathbf{x}_t)\, \sigma_\mathrm{t}(\mathbf{x}_t) \Big(\sum_i a^i\, v^i(\mathbf{x}_t) \Big)$. Hence, each entry in $\mathbf{W}\mathbf{a}$ can be determined via rendering using a volume defined by $\mathbf{a}$. Next, computing $\mathbf{c} := (\ldots, c^k, \ldots)^\mathrm{T} := \mathbf{W}\mathbf{a} - \mathbf{o}$ is straightforward, but applying $\mathbf{W}^\mathrm{T}$ to $\mathbf{c}$ is a data scattering operation. Each value c^k is associated to a constraint pixel k. It needs to be scattered to those voxels that are traversed by a ray marching procedure for the ray associated with pixel k, weighted according to the voxel's influence on k. In other words, we have to perform a ray marching that is very similar to the case for computing the product with $\mathbf{W}$.

In fact, both operations are so similar, that almost the same code can be reused. During rendering, which implements the multiplication by $\mathbf{W}$, we use ray-marching to sum the voxel contributions along the ray following $\mathrm{outRadiance} \mathrel{+}= \mathrm{weight} \cdot \mathrm{texRead}(\mathbf{a}, \mathrm{rayPos})$. The variable weight includes the transmittance between the current position rayPos and the ray origin due to the volumes' extinction, as well as the value of the basis function at the current position. To implement the multiplication by $\mathbf{W}^\mathrm{T}$, this single line

is exchanged by $\mathrm{texWriteAdd}(\mathbf{a}, \mathrm{rayPos}, \mathrm{weight} \cdot c^k)$. Reusing the code is also beneficial as the weight values match up perfectly, and, consequently, the implicitly constructed matrices $\mathbf{W}$ and $\mathbf{W}^{\mathrm{T}}$ agree with each other. The full pseudo code can be found in the supplementary material.

Optimizations are possible to speed up the rendering shaders: while multiplying with $\mathbf{W}$ is efficient and directly parallelized as usual when stepping through the volume per pixel, the situation is different for multiplication by $\mathbf{W}^{\mathrm{T}}$. As we scatter data (texture writes are realized via `shader_image_load_store`), synchronization issues may occur. Rays of neighboring pixels will likely write to the same voxel. To avoid the resulting stall, we use an interleaved pattern of 6×6 pixels. In each round, only one ray of these sub-windows is shot, decreasing the number of conflicts and speeding up the computation by $\approx 10\%$. Further, the use of 16bit instead of 32bit textures leads to a speedup of $\approx 25\%$.

5.1 Optimization Extensions

Higher precision is obtained when using accurate ray traversal [Amanatides and Woo 1987] instead of marching. For several basis-function choices, e.g., nearest-neighbor or linear (which we use), an accurate integral can be computed. This applies for the accumulated transmittance value during volume traversal as well as the weight of the basis function itself.

Regularization is a standard way of stabilizing and controlling the optimization. Usually, additional quadratic terms, modeling prior knowledge about the solution space, are added to the quadratic error function to address numerical ill-conditioning. A Laplacian term $\|\mathrm{L}\mathbf{a}\|^2$ can smooth the overall volume. $\|\mathbf{a}\|^2$ minimizes the solution, while $\|\mathbf{a} - 1\|^2$ biases it towards one. Each of these regularizers is controlled by user-defined weights. Thus, when optimizing for albedo and emission, we can specify which element to favor and, e.g., minimize emission. In practice, we use weak weights (i.e., 2^{-9}), which proved sufficient for a stable solution.

Weights are often desirable to give different parts of an image more importance than others. They are also handy when creating transitions, e.g., when editing only a part of the volume. These per-constraint pixel weights can be easily integrated into the conjugate gradient method and need to be multiplied with the vector $\mathbf{W}\mathbf{a} - \mathbf{o}$.

Refraction occurs for non-constant refractive indices and rays are bent during the traversal. Our reconstruction scheme can handle arbitrary integration curves c of known geometry (cf. Eq. 1), which enables us to use more complex paths than a straight line. To compute the refractive ray paths in our volumetric setting, we resort to the Euler forward scheme of [Ihrke et al. 2007].

Visual Hulls can be used to limit the domain of solution [Ihrke and Magnor 2004], which increases quality and performance. In case of emission/albedo optimization, we only consider voxels with non-zero extinction, as the other areas have no influence on the rendering. For estimating extinction itself, the input views of the user can be transformed into masks. In all cases, the user can specify masks (projections of the desired visual hull) as additional input to the optimization. Those masks need not to align with any of the input images, the parts of the volume that are not inside the visual hull are simply ignored during the optimization.

2 views 512x512 pixels each	volume resolution 32^3	64^3	128^3
ray marching steps / diagonal 64	78	108	219
128	106	141	259
256	159	205	334
512	262	331	487

steps/diag.	64	128	256
vol. resolution	32^3	64^3	128^3
# of views 512x512 pixels each 2	76	139	330
4	129	229	523
6	184	326	712
8	237	418	897

Figure 4: *Timings in ms for Bunny per iteration (three for convergence). Left: volume resolution vs. ray marching steps (relative to volume diagonal), more steps estimate the integrals more accurately. Right: volume resolution vs. # of views. In addition to the number of inputs, the performance of the optimization depends on volume resolution as well as the number of ray marching steps.*

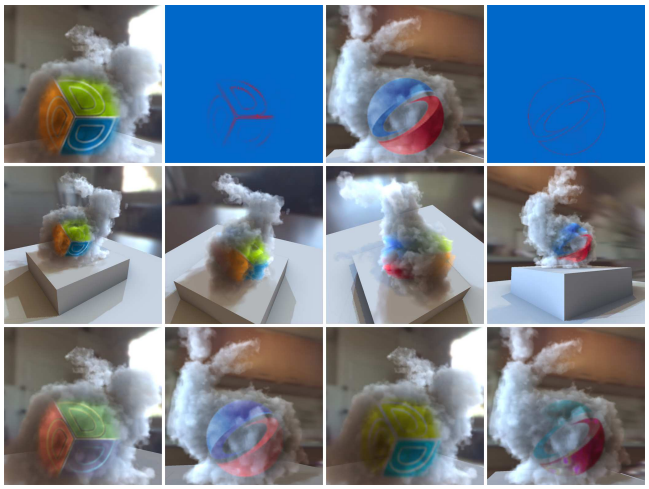

Figure 5: *Logos on a smoke bunny. Top: optimized emission and albedo and $8x$ difference (false color: blue to red); Middle: Intermediate views appear plausible. Bottom: left two images use only emission, the other two only albedo.*

6 Results and Discussion

We used an Intel Xeon x5650 with NVIDIA 560Ti. We obtain fast execution times, even for medium sized voxel grids and high precision. The execution times for different numbers of ray marching steps, voxel resolution, and input images are given in Fig. 4. The process converges after three to five conjugate gradient steps. For 128^3 voxels, 256 ray marching steps, 524K constraint pixels (2 views), the result (Fig. 5) is computed in less than two seconds. With 2.1M constraint pixels (9 views) the computation takes seven seconds (Fig. 6). In practice, a good choice is twice the number of ray marching steps for the length of the diagonal as the number of voxels along an axis. In the following, we illustrate our method on three examples. It took a user roughly 2 min to produce the smoke bunny, 15 minutes for the cloud scene and 4 min for the refraction.

Smoke Bunny shows our approach with imprinted logos. These were blended with the original appearance in the user-provided views - a purely artistic choice. When optimizing albedo and emission, the result is a close match to the input (Fig. 5, top), while intermediate views look appealing and plausible (middle). One can also modify emission or albedo only (bottom). The result can no longer match the input. Emission can only lighten, albedo only darken the appearance (following physical constraints). Depending on the required realism and desired material, these could be adequate choices.

Figure 6: *Cloud stylization; original volume (left), user input (middle), result after optimization (right), except for the last row, which shows random intermediate views.*

Figure 7: *Single (left) vs. multiple scattering (right). To account for the additional energy, we scaled the environmental light such that both images have similar brightness.*

Cloud Stylization illustrates the expressiveness of our system. The scene shows a cloud over a city and the goal is to achieve a "frightening" look when on one side of the cloud, a calm appearance on the other side. Modifying appearance in 10 views (640×360 pixel), lead to the result in Fig. 6.

The number of views is slightly increased for two reasons; to define a good appearance all around the object and to avoid visual inconsistencies. The system only optimizes for parts of the volume that are specified in at least a single view. Other parts are unchanged and may result in a soft *edge* after the optimization. Also, if insufficient constraints are used, stripes can become visible, as the color of a pixel constraint only defines the voxels along its corresponding ray. One could increase the weight of the Laplacian regularizer to avoid these artifacts, but we found that this unnecessarily destroys much of the intricate detail.

The view modifications were done by contrast enhancement in the according views in conjunction with a blended red mask. The resulting input images were *inconsistent* in parts, as each was independently designed. Yet, the least-square solution maintains the most faithful reconstruction.

The effect of rendering the stylized volume under multiple scattering is shown in Fig. 7. While multiple scattering acts like a blur, that reduces details, the reconstructed logo remains clearly visible.

Extinction and Refraction is shown in Fig. 8. Here, we illustrate the use of our extinction optimization to show the construction of complex shapes and also to illustrate the compatibility with bent rays due to continuous refraction. The user provided four input views (growth of a tree) and our reconstruction computed extinction coefficients (per wavelength) that attenuate the background light such as to match the provided images. In the example, the volume does not scatter or emit any light, i. e., $Q_o = 0$. In this reconstruction process, the shape of the volume is implicitly defined by the volume of varying refractive index. The refraction indices were generated using a random process restricted to the inside of a glass sphere. We also used per-pixel weights to concentrate the importance on the main parts of the user images, which leads to a less constrained boundary and a more variable shape. Due to the refraction, the intermediate views appear as random colors. We imagine, such a combination of refraction and extinction could e.g. be used in a game, where the user has to find the correct viewpoint to see a certain image, in order to obtain hints for the solution of a puzzle.

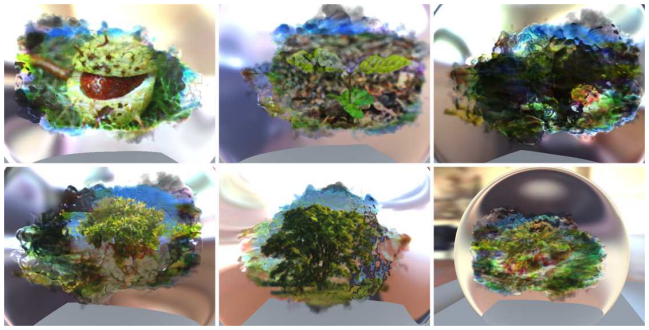

Figure 8: *Reconstruction of extinction in a refractive volume to absorb light of the background. Left, middle: results for user-defined views, rightmost column: intermediate views.*

7 Conclusions

We presented a novel approach to stylize volumes. We showed that solving for a specific volume appearance is possible by adjusting the desired appearance in a number of reference views. Our formulation as an optimization problem results in a large scale linear system. We make this solution practical by employing an efficient GPU-friendly implementation that avoids the explicit construction of the equation system. Several cases illustrated the variability of the method.

In the future, we plan to consider multiple scattering, the current problem being the cost of derivative computations and the demands on memory. Fully GPU-based out-of-core rendering approaches hint at a possible direction [Crassin et al. 2010].

Acknowledgements

We would like to thank Bernhard Reinert, and colleagues at MPI for helpful discussions and the reviewers for their valuable feedback. This work was supported by the German Research Foundation (DFG) through the Emmy-Noether fellowship IH 114/1-1 and by the Intel Visual Computing Institute at Saarland University. The city model is from 3DRT.

References

AMANATIDES, J., AND WOO, A. 1987. A Fast Voxel Traversal Algorithm for Ray Tracing. In *Proc. Eurographics*, 3–10.

ATCHESON, B., IHRKE, I., HEIDRICH, W., TEVS, A., BRADLEY, D., MAGNOR, M., AND SEIDEL, H.-P. 2008. Time-resolved

3d capture of non-stationary gas flows. *ACM Trans. Graphics (Proc. Siggraph ASIA) 27*, 5.

BARAN, I., KELLER, P., BRADLEY, D., COROS, S., JAROSZ, W., NOWROUZEZAHRAI, D., AND GROSS, M. 2012. Manufacturing Layered Attenuators for Multiple Prescribed Shadow Images. *Computer Graphics Forum 31*, 2, 603–610.

CHANDRASEKAR, S. 1960. *Radiative Transfer*. Dover Publications.

CRASSIN, C., NEYRET, F., SAINZ, M., AND EISEMANN, E. 2010. *GPU Pro*. AK Peters, ch. X.3 Efficient Rendering of Highly Detailed Volumetric Scenes with GigaVoxels, 643–676.

FUCHS, M., RASKAR, R., SEIDEL, H.-P., AND LENSCH, H. P. A. 2008. Towards Passive 6D Reflectance Field Displays. *ACM Trans. Graph. 27*, 3, 58:1–58:8.

HAWKINS, T., EINARSSON, P., AND DEBEVEC, P. 2005. Acquisition of Time-Varying Participating Media. In *Proc. SIGGRAPH*, 812–815.

HOLROYD, M., BARAN, I., LAWRENCE, J., AND MATUSIK, W. 2011. Computing and fabricating multilayer models. *ACM Trans. Graph. 30*, 6, 187:1–187:8.

IHRKE, I., AND MAGNOR, M. 2004. Image-Based Tomographic Reconstruction of Flames. *Proc. SCA*, 367–375.

IHRKE, I., AND MAGNOR, M. 2006. Adaptive Grid Optical Tomography. *Graphical Models 68*, 484–495.

IHRKE, I., GOLDLUECKE, B., AND MAGNOR, M. 2005. Reconstructing the Geometry of Flowing Water. In *Proc. ICCV*, 1055–1060.

IHRKE, I., ZIEGLER, G., TEVS, A., THEOBALT, C., MAGNOR, M., AND SEIDEL, H.-P. 2007. Eikonal Rendering: Efficient Light Transport in Refractive Objects. *ACM Trans. Graph. 26*, 3, 59:1–59:8.

JAROSZ, W., NOWROUZEZAHRAI, D., THOMAS, R., SLOAN, P.-P., AND ZWICKER, M. 2011. Progressive Photon Beams. *ACM Trans. Graph. 30*, 6.

KAJIYA, J., AND VON HERZEN, B. 1984. Ray Tracing Volume Densities. In *Proc. SIGGRAPH*, 165–174.

KAK, A. C., AND SLANEY, M. 1988. *Principles of Computerized Tomographic Imaging*. IEEE Press.

KERR, W. B., PELLACINI, F., AND DENNING, J. D. 2010. BendyLights: Artistic Control of Direct Illumination by Curving Light Rays. *Comput. Graph. Forum 29*, 4, 1451–1459.

LANMAN, D., WETZSTEIN, G., HIRSCH, M., HEIDRICH, W., AND RASKAR, R. 2011. Polarization Fields: Dynamic Light Field Display using Multi-Layer LCDs. *ACM Trans. Graph. 30*, 6.

LINŢU, A., LENSCH, H. P. A., MAGNOR, M., EL-ABED, S., AND SEIDEL, H.-P. 2007. 3D Reconstruction of Emission and Absorption in Planetary Nebulae. In *Proc. Volume Graphics*, 9–16.

MAX, N. 1995. Optical Models for Direct Volume Rendering. *IEEE TVCG 1*, 2, 99–108.

MCNAMARA, A., TREUILLE, A., POPOVIĆ, Z., AND STAM, J. 2004. Fluid Control using the Adjoint Method. In *Proc. SIGGRAPH*, 449–456.

NOVÁK, J., NOWROUZEZAHRAI, D., DACHSBACHER, C., AND JAROSZ, W. 2012. Progressive Virtual Beam Lights. *Computer Graphics Forum 31*, 4, 1407–1413.

NOWROUZEZAHRAI, D., JOHNSON, J., SELLE, A., LACEWELL, D., KASCHALK, M., AND JAROSZ, W. 2011. A Programmable System for Artistic Volumetric Lighting. *ACM Trans. Graph. 30*, 4, 29:1–29:8.

OBERT, J., KRIVÁNEK, J., PELLACINI, F., SÝKORA, D., AND PATTANAIK, S. N. 2008. iCheat: A Representation for Artistic Control of Indirect Cinematic Lighting. *Comput. Graph. Forum 27*, 4, 1217–1223.

OBERT, J., PELLACINI, F., AND PATTANAIK, S. N. 2010. Visibility Editing For All-Frequency Shadow Design. *Comput. Graph. Forum 29*, 4, 1441–1449.

PAPAS, M., JAROSZ, W., JAKOB, W., RUSINKIEWICZ, S., MATUSIK, W., AND WEYRICH, T. 2011. Goal-based Caustics. *Computer Graphics Forum 30*, 2, 503–511.

PAPAS, M., HOUIT, T., NOWROUZEZAHRAI, D., GROSS, M., AND JAROSZ, W. 2012. The magic lens: Refractive steganography. *ACM Trans. Graph. 31*, 6.

PAULY, M., AND MITRA, N. 2009. Shadow Art. *ACM Trans. Graph. 28*, 5, 156:1–156:7.

PERLIN, K. 1989. Hypertexture. In *Proc. SIGGRAPH*, 253–262.

SCHOENEMAN, C., DORSEY, J., SMITS, B. E., ARVO, J., AND GREENBURG, D. 1993. Painting with Light. In *Proc. SIGGRAPH*, 143–146.

SHEWCHUK, J. R. 1994. An introduction to the conjugate gradient method without the agonizing pain. Tech. rep., Pittsburgh, PA, USA.

SONG, Y., TONG, X., PELLACINI, F., AND PEERS, P. 2009. SubEdit: a Representation for Editing Measured Heterogeneous Subsurface Scattering. 31:1–31:10.

STAM, J. 1999. Stable Fluids. In *Proc. SIGGRAPH*, 121–128.

TREUILLE, A., MCNAMARA, A., POPOVIĆ, Z., AND STAM, J. 2003. Keyframe Control of Smoke Simulations. In *Proc. SIGGRAPH*, 716–723.

WETZSTEIN, G., LANMAN, D., HEIDRICH, W., AND RASKAR, R. 2011. Layered 3D: Tomographic Image Synthesis for Attenuation-based Light Field and High Dynamic Range Displays. *ACM Trans. Graph. 30*, 4, 95:1–95:12.

WETZSTEIN, G., LANMAN, D., HIRSCH, M., AND RASKAR, R. 2012. Tensor Displays: Compressive Light Field Synthesis using Multilayer Displays with Directional Backlighting. *ACM Trans. Graph. 31*, 4, 1–11.

WEYRICH, T., PEERS, P., MATUSIK, W., AND RUSINKIEWICZ, S. 2009. Fabricating Microgeometry for Custom Surface Reflectance. *ACM Trans. Graph. 28*, 3, 32:1–32:6.