



HAL
open science

A new microflake model with microscopic self-shadowing for accurate volume downsampling

Guillaume Loubet, Fabrice Neyret

► To cite this version:

Guillaume Loubet, Fabrice Neyret. A new microflake model with microscopic self-shadowing for accurate volume downsampling. *Computer Graphics Forum*, 2018, 37 (2), pp.111-121. 10.1111/cgf.13346 . hal-01702000

HAL Id: hal-01702000

<https://hal.science/hal-01702000>

Submitted on 6 Feb 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A new microflake model with microscopic self-shadowing for accurate volume downsampling

Guillaume Loubet[†] and Fabrice Neyret[†]

INRIA, Univ. Grenoble Alpes/LJK, CNRS/LJK

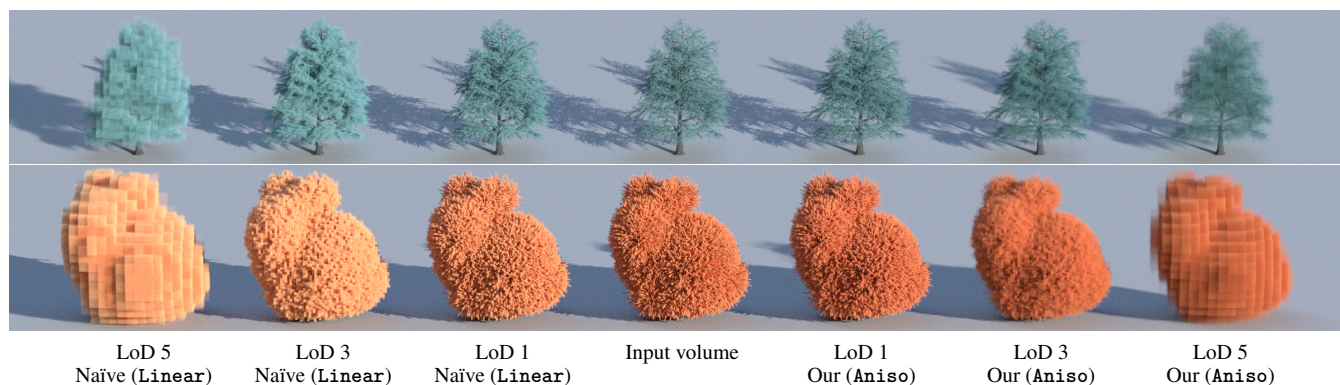


Figure 1: Comparison between naïve downsampling of microflake volumes and our method (**Aniso**, Sec. 7). Naïve downsampling of dense heterogeneous volumes often lead to inaccurate LoDs, due to the loss of masking and shadowing effects that occur between and inside dense input voxels. Our downsampling approach is based on a new participating medium model and on local estimations of self-shadowing probabilities. It generates LoDs with correct transparency and consistent appearance through scales. Rendered with volume path tracing (the trunk of the cedar is a mesh).

Abstract

Naïve linear methods for downsampling high-resolution microflake volumes often produce inaccurate appearance, especially when input voxels are very opaque. Preserving correct appearance at all resolutions requires taking into account masking-shadowing effects that occur between and inside dense input voxels. We introduce a new microflake model whose additional parameters characterize self-shadowing effects at a microscopic scale. We provide an anisotropic self-shadowing function and microflake distributions for which the scattering coefficients and the phase functions of our model have closed-form expressions. We use this model in a new downsampling approach in which scattering parameters are computed from local estimations of self-shadowing probabilities in the input volume. Unlike previous work, our method handles datasets with spatially varying scattering parameters, semi-transparent volumes and datasets with intricate silhouettes. We show that our method generates LoDs with correct transparency and consistent appearance through scales for a wide range of challenging datasets, allowing for huge memory savings and efficient distant rendering without loss of quality.

CCS Concepts

•Computing methodologies → Ray tracing; Volumetric models;

1. Introduction

Heterogeneous participating media based on voxel grids are a powerful representation for rendering complex semi-transparent appearance and intricate shapes. Rendering high-resolution volume

data is challenging due to high I/O time [ZWDR16] and costly volume integration along rays, but contrary to mesh-based intricate geometry, voxel grids are convenient for level-of-details (LoDs) and allow for efficient rendering and low memory usage when an appropriate resolution is used. Unfortunately, downsampling volume data is not as straightforward as it seems since naïve downsampling methods based on linear pre-filtering often produce inconsistent ap-

[†] guillaume.loubet@gmx.fr & fabrice.neyret@inria.fr

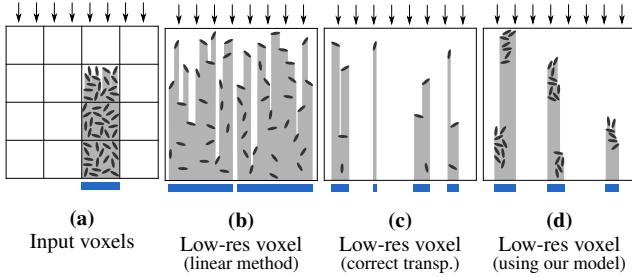


Figure 2: In the case of a highly heterogeneous volume (a), naïve downsampling algorithms can lead to inaccurate results (b-c). (b): linear pre-filtering densities results in incorrect transparency (blue lines) in low-resolution voxels. (c): correct transparency can be obtained by decreasing density in low-resolution voxels, but this also decreases the probability of local self-shadowing and leads to overly bright LoDs. (d): Our microscopic self-shadowing model can represent media with correlated microflakes and it preserves both anisotropic transparency and self-shadowing effects.

pearance through scales as shown in Fig. 1 (left), especially when input datasets have dense voxels. Indeed, linear pre-filtering does not preserve accurate transparency as shown in Fig. 2b, and decreasing density in low-resolution voxels necessarily reduces the amount of local *self shadowing and masking* [ZWDR16], that is the probability that light scattered from a point in the medium is scattered again in its close vicinity. This loss of self-shadowing is illustrated in Fig. 2c. The lack of local multiple scattering leads to overly bright LoDs when naïve downsampling methods are used (Fig. 1, left).

Among available participating medium models in Computer Graphics, the microflake model [JAM*10] has been used for rendering anisotropic fiber-like and foliage-like appearance. Recently, two publications overcame important problems raised by microflake volume downsampling. Heitz et al. [HDCD15] have introduced the SGGX microflake distribution which support efficient linear pre-filtering. Zhao, Wu et al. [ZWDR16] have built on this work and have introduced a downsampling approach that preserves correct appearance in LoDs. As discussed in Section 2.3, their approach has some limitations due to the use of large-scale training rendering for optimizing scattering parameters and voxel clustering that produces artifacts when datasets have spatially varying appearance. In this paper, we introduce a new downsampling approach for generating appearance-preserving LoDs. Our main contributions include:

- A new microflake model whose parameters characterize self-shadowing effects at a microscopic scale (Sec. 4).
- A new downsampling algorithm that uses our model for preserving both correct transparency and self-shadowing effects in low-resolution volumes (Sec. 7).
- We derive closed-form expressions and sampling procedures for implementing our model and its simplified version (Sec. 4.3), using appropriate self-shadowing functions and microflake distributions (Sec. 5 and 6).

Our downsampling algorithm computes scattering parameters in low-res voxels from local estimations of self-shadowing probabilities in input voxels. This approach overcomes some limitations of previous work [ZWDR16]: it allows for downsampling without artifacts spatially varying datasets, because it does not require voxel clustering, and it supports semi-transparent datasets because it avoids the optimization of parameters in image space (Sec. 2.3). Our downsampling approach is also faster because local estimations are less expensive than large-scale rendering (Sec. 8.1).

Our new microflake model aims at adjusting anisotropic attenuation and self-shadowing in low-resolution voxels. Our phase functions do not model accurately scattering that occurs in very dense voxels, which limits accuracy of our LoDs. Preserving both accurate opacity and phase functions for dense heterogeneous volumes remains an open problem as discussed in Sec. 8.4. In this work, we did not focus on fabrics: unlike Zhao, Wu et al. [ZWDR16], we do not address the case of multi-yarn datasets with strong correlations between albedos and microflake distributions.

2. Related work

2.1. Volume scattering models in computer graphics

Most participating medium models in computer graphics are based on the standard radiative transfer framework [Cha50]. Several extensions have been proposed such as discrete media approximations [MWM07, MPH*15, MPG*16] and models for media with anisotropic attenuation and direction-dependent phase functions [Ney98, MWM08, SKZ11, KSZ*15]. Jakob et al. [JAM*10] have generalized the radiative transfer equation to such anisotropic media and have introduced the physically-based microflake model. This model has been used for rendering and pre-filtering intricate shapes such as foliage [HDCD15, LN17] and fabrics [ZJMB11, ZWDR16]. In our work, we have focused on downsampling microflake volumes, and we believe that our approach can be adapted to other volume models, e.g. simpler isotropic volume models.

2.2. Normal distributions in the microflakes framework

The key ingredient of the microflake model is the microflake normal distribution function, from which phase functions and attenuation coefficients are derived. Rendering with the microflake model requires efficient evaluation of the projected area of microflakes as well as efficient evaluation and sampling procedures for phase functions. We briefly review here existing microflake normal distribution functions because choosing appropriate distributions has been essential in our work for implementing our new microflake model (Sections 5 and 6).

In the seminal work of Jakob et al. [JAM*10], the authors have proposed distributions based on powers of sine and cosine functions: $D_{\text{surf}}(\omega) = \cos^n(v, \omega)$ and $D_{\text{fiber}}(\omega) = \sin^n(v, \omega)$. They have not provided closed-form expressions for attenuation coefficients and they have used spherical harmonic approximations for evaluation and sampling. In our work, we combine similar distributions because of their good mathematical properties (Sec. 5). We

provide the missing closed-form expressions for attenuation coefficients and phase functions, as well as exact importance sampling procedures.

Zhao et al. [ZJMB11] have used another distribution called the Gaussian Fiber Distribution. Their implementation is more efficient than the original implementation of Jakob et al. [JAM*10], although they have not provided closed-form expression of the attenuation coefficient – but they provided an efficient approximation [Jak10]. They have used a rejection algorithm for sampling the distribution of visible normals [HDCCD15] based on the sampling of the normal distribution, which is inefficient in some configurations. We also rely on rejection sampling for some of our phase functions but we provide robust sampling procedures for the distributions of visible normals.

Heitz et al. [HDCCD15] have introduced the SGGX distribution. It has overcome limitations of previous distributions since it supports fast and exact evaluation of attenuation coefficients, as well as efficient evaluation and sampling of specular phase functions. We have used the SGGX distribution for implementing our simplified self-shadowing model (Section 6).

2.3. Downsampling volume data

Several authors have proposed volume downsampling methods in various contexts including simple RGBA data [KB08] and anisotropic voxels for real-time global illumination [CNS*11]. Heitz et al. have addressed the problem of downsampling microflakes normal distributions [HDCCD15] and Zhao, Wu et al. [ZWDR16] have built on this work for downsampling microflakes volumes taking into account shadowing effects. They have proposed an iterative optimization of scattering parameters using large-scale training rendering. The optimization stops when appearance of LoDs matches the reference volume. This approach has proven to produce accurate LoDs for several challenging datasets including complex multi-fibers fabrics. However, the enormous amount of parameters requiring optimization makes the problem intractable unless low-resolution voxels are assigned to a few clusters that share some scattering parameters. Voxel clustering reduces dimensionality but leads to artifacts when input dataset have spatially varying scattering parameters. Another limitation of their work is that their optimization using image space errors assumes that each pixel value depends mostly on the first non-empty voxels that are seen directly. It is unclear how their algorithm could be adapted to semi-transparent datasets, or datasets that have both transparent and very dense voxels.

In this paper, we address the same problem as Zhao, Wu et al. [ZWDR16], that is generating low-resolution microflake volume while preserving the appearance at all scales. We add the constraint that our LoDs must have correct transparency and silhouettes, and we do not rely on linear downsampling for density parameters. Correct transparency is important when downsampling intricate datasets such as the cedar in Fig. 1. We overcome problems raised by voxel clustering and optimization with image space

| Symbol | Meaning | Unit |
|--------------------------------------|--------------------------------------|-----------|
| $D(\omega)$ | Microflake normal distribution | st^{-1} |
| $A(\omega)$ | Microscopic self-shadowing | - |
| ρ | Total area of flakes per unit volume | m^{-1} |
| σ_t | Attenuation coefficient | m^{-1} |
| $\sigma_s, \sigma_{ss}, \sigma_{ms}$ | Scattering coef. (wavelength dep.) | m^{-1} |
| f, f_{ss}, f_{ms} | Normalized phase functions | st^{-1} |
| $\alpha, \alpha_{ss}, \alpha_{ms}$ | Albedos (wavelength dep.) | - |
| (\cdot) | Dot product | - |
| $\langle \cdot \rangle$ | Clamped dot product | - |

Table 1: Main symbols used in this paper.

errors, using local estimations of self-shadowing probabilities instead of large-scale training rendering.

3. Background: the standard microflake model

Jakob et al. [JAM*10] have introduced a physically-based model in which the medium is, at a microscopic scale, a homogeneous cloud of microflakes whose orientations are statistically described by a microflake normal distribution function. Anisotropic microflake normal distributions result in anisotropic attenuation coefficients and direction-dependent phase functions.

Our self-shadowing model (Sec. 4) is an extension of the standard microflake model, in which we added correlations of microflakes position at a microscopic scale (Fig. 2d). In this section, we briefly review the anisotropic radiative transfer equation introduced by Jakob et al. [JAM*10] and their microflake model, which will be referred to as the *standard microflake model* in this paper.

Anisotropic RTE. The anisotropic radiative transfer equation (RTE) proposed by Jakob et al. [JAM*10] writes:

$$(\omega \cdot \nabla)L(\omega) + \sigma_t(\omega)L(\omega) = \sigma_s(\omega) \int_{S^2} f(\omega \rightarrow \omega')L(\omega')d\omega' + Q(\omega)$$

with $\sigma_t(\omega)$ the anisotropic attenuation coefficient, $\sigma_s(\omega)$ the anisotropic scattering coefficient (which is usually wavelength dependent), and f the anisotropic phase function (in the sense that it depends on ω and ω' , and not only on the angle between ω and ω'). In this paper, we use phase functions that are normalized over the second parameter:

$$\int_{S^2} f(\omega \rightarrow \omega')d\omega' = 1, \quad \forall \omega.$$

Helmholtz's reciprocity principle for anisotropic media. In general, phase functions in the anisotropic RTE framework do not satisfy $f(\omega \rightarrow \omega') = f(\omega' \rightarrow \omega)$. The Helmholtz's reciprocity principle, which states that radiative transfer remains the same when interchanging sources and receivers, writes

$$\sigma_s(\omega)f(\omega \rightarrow \omega') = \sigma_s(\omega')f(\omega' \rightarrow \omega).$$

This is the reason why using arbitrary phase functions for medium with anisotropic attenuation coefficient often break reciprocity.

Microflake model. Based on their anisotropic RTE, Jakob et al. [JAM*10] have studied the case of medium made of microflakes, given a microflake normal distribution D . They derived attenuation coefficients and phase functions for such media and the Helmholtz's reciprocity principle is satisfied by construction. The microflake BRDF is often considered perfectly specular because this greatly simplifies evaluation and sampling.

In this work, we assume like Heitz et al. [HDCD15] that microflakes reflect on both side and we only consider normal distributions D that satisfy

$$D(\omega) = D(-\omega).$$

Given D , the attenuation coefficient is derived from the integral of projected areas over microflake orientations:

$$\sigma_t(\omega) = \rho \int D(\omega_m) \langle \omega \cdot \omega_m \rangle d\omega_m \quad (3.1)$$

where ρ , which will be referred to as *density* in this paper, is the amount of microflake surface per unit volume (Table 1). Assuming that the microflake albedo is not view-dependent, the scattering coefficient for a wavelength λ is given by

$$\sigma_s(\omega) = \rho \alpha(\lambda) \int D(\omega_m) \langle \omega \cdot \omega_m \rangle d\omega_m = \alpha(\lambda) \sigma_t(\omega) \quad (3.2)$$

with $\alpha(\lambda)$ the albedo of the microflakes. In the case of specular microflakes, the phase function writes

$$f(\omega \rightarrow \omega') = \frac{\rho \alpha(\lambda) D(\omega_h)}{4\sigma_s(\omega)} \quad (3.3)$$

with $\omega_h = \frac{\omega + \omega'}{\|\omega + \omega'\|}$ the half-vector.

4. The microscopic self-shadowing microflake model

In this section, we introduce a new participating medium model based on the microflake model. We call it the *microscopic self-shadowing model* because it models media whose microflakes have correlated positions at a microscopic scale, leading to microscopic shadowing and masking. Microscopic self-shadowing impacts attenuation coefficients, as shown in Fig. 2, as well as scattering coefficients and phase functions.

4.1. Motivations for a new model

Volume downsampling raises the problem of how to best represent a set of heterogeneous input voxels with a single low-resolution voxel. The approach of Zhao, Wu et al. consists in using standard microflake volumes and optimizing microflake albedos in low-res voxels in order to take into account self-shadowing. Instead, we introduce a new microflake model whose parameters allow:

- the control of the anisotropic attenuation due to anisotropic correlations, independently of the microflake normal distribution (Fig. 2d),

- the control of the amount of microscopic self-shadowing, using different phase functions and albedos for single scattering and multiple scattering at the microscopic scale.

4.2. Our model

Let's consider a volume whose microflakes are spatially correlated at a microscopic scale, as shown in Fig. 2d. We characterize microflakes correlations with the probability that a microflake is shadowed or masked by a neighboring microflake, in a given direction. We introduce a dimensionless scalar function A on the sphere such that $1 - A(\omega)$ gives the probability of shadowing by other neighboring microflakes in the direction ω . A satisfies:

$$\forall \omega \in S^2, 0 < A(\omega) \leq 1 \text{ and } A(\omega) = A(-\omega). \quad (4.1)$$

The case $A(\omega) = 1$ corresponds to the standard microflake model in which microflakes are well separated at a microscopic scale and not correlated [JAM*10]. Given this shadowing behavior at a microscopic scale, we derive new expressions for attenuation coefficients, scattering coefficients and phase functions.

Attenuation coefficient (σ_t). The attenuation coefficient is given by the standard microflake model times the self-shadowing function A characterizing microflake correlations:

$$\sigma_t(\omega) = A(\omega) \rho \int D(\omega_m) \langle \omega \cdot \omega_m \rangle d\omega_m. \quad (4.2)$$

Single scattering coefficient (σ_{ss}). We consider the amount of *local single scattering* $\sigma_{ss}(\omega)$ at microscopic scale, from an incoming direction ω . Single scattering occurs when microflakes are both unmasked and un-shadowed:

$$\sigma_{ss}(\omega) = \rho \alpha_{ss}(\lambda) A(\omega) \int A(\omega') D(\omega_m) \langle \omega \cdot \omega_m \rangle d\omega_m \quad (4.3)$$

where $\omega' = 2\omega_m(\omega_m \cdot \omega) - \omega$ is the reflected direction given an input direction ω and a microflake normal ω_m . This can be written as an integral over outgoing directions, using the Jacobian of the transformation from normals to specular reflections provided by Walter et al. [WMLT07]:

$$\sigma_{ss}(\omega) = \frac{\rho \alpha_{ss}(\lambda) A(\omega)}{4} \int A(\omega') D(\omega_h) d\omega'. \quad (4.4)$$

The single scattering phase function (f_{ss}). Microscopic self-shadowing also impacts the *single scattering phase function* f_{ss} , which is the standard phase function attenuated in some directions due to microscopic shadowing (and normalized):

$$\begin{aligned} f_{ss}(\omega \rightarrow \omega') &= \frac{A(\omega') D(\omega_h)}{\int A(\omega'') D((\omega'' + \omega) / \|\omega'' + \omega\|) d\omega''} \\ &= \frac{\rho \alpha_{ss}(\lambda) A(\omega) A(\omega') D(\omega_h)}{4\sigma_{ss}(\omega)}. \end{aligned} \quad (4.5)$$

Again, when $A = 1$, this reduces to the standard microflake specular phase function. It is easy to check that our model satisfies Helmholtz's reciprocity principle by construction:

$$\sigma_{ss}(\omega) f_{ss}(\omega \rightarrow \omega') = \frac{\rho \alpha_{ss}(\lambda) A(\omega) A(\omega') D(\omega_h)}{4}$$

which is symmetric in ω and ω' .

Microscopic multiple scattering. We want our model to take into account microscopic multiple scattering so that it satisfies the *white furnace test*, meaning that all incoming light should be scattered in the case of microflakes with albedo 1, for any self-shadowing function A .

Ideally, our model should have one scattering coefficient and one phase function for each number of local scattering events. Unfortunately, their expressions involve spherical convolutions of the microflake phase function, for which there are no analytic expressions in general. In this paper, we chose to use a single multiple scattering coefficient $\sigma_{ms}(\omega)$ and a single multiple scattering albedo $\alpha_{ms}(\lambda)$ that approximate contributions of all scattering orders except single scattering. The multiple scattering coefficient is given by the amount of light that scattered and then masked locally, times the multiple scattering albedo:

$$\begin{aligned}\sigma_{ms}(\omega) &= \alpha_{ms}(\lambda) \rho A(\omega) \int (1 - A(\omega')) D(\omega_m) \langle \omega \cdot \omega_m \rangle d\omega_m \\ &= \alpha_{ms}(\lambda) \left(\sigma_t(\omega) - \frac{\sigma_{ss}(\omega)}{\alpha_{ss}(\lambda)} \right).\end{aligned}\quad (4.6)$$

The multiple scattering albedo $\alpha_{ms}(\lambda)$ is a parameter of our model. We assume that multiple scattering is roughly diffuse, and we use the multiple scattering phase function

$$f_{ms}(\omega \rightarrow \omega') = f_{ms}(\omega') = \frac{\sigma_{ms}(\omega')}{\int \sigma_{ms}(\omega'') d\omega''}.\quad (4.7)$$

It satisfies reciprocity since

$$\sigma_{ms}(\omega) f_{ms}(\omega \rightarrow \omega') = \frac{\sigma_{ms}(\omega) \sigma_{ms}(\omega')}{\int \sigma_{ms}(\omega'') d\omega''}$$

which is symmetric in ω and ω' . This phase function scatters light in directions in which there is strong microscopic shadowing. Also note that in the case $\alpha_{ss}(\lambda) = \alpha_{ms}(\lambda) = 1$, we have

$$\sigma_{ss}(\omega) + \sigma_{ms}(\omega) = \sigma_{ss}(\omega) + \alpha_{ms}(\lambda) \left(\sigma_t(\omega) - \frac{\sigma_{ss}(\omega)}{\alpha_{ss}(\lambda)} \right) = \sigma_t(\omega)$$

meaning that our model preserves energy when microflakes do not absorb light, meaning that the model passes the white furnace test.

Summary. We introduced a self-shadowing function A , where $1 - A(\omega)$ represents the probability of shadowing and masking by neighboring microflakes at a microscopic scale. Self-shadowing impacts attenuation coefficient σ_t , the single scattering coefficient σ_{ss} and the associated phase function f_{ss} . For energy conservation, we take into account local multiple scattering introducing a multiple scattering coefficient σ_{ms} and its associated phase function f_{ms} . Rendering with this model requires evaluating these functions (σ_t , σ_{ss} , f_{ss} , σ_{ms} , f_{ms}) and sampling procedures for f_{ss} and f_{ms} .

In practice, Eq. 4.2, 4.3 and 4.7 rely on spherical integrals of the microflakes distribution D with the shadowing function A , meaning that for most functions D and A no closed-form expressions can be found. In the next section, we present appropriate functions D and A that lead to closed-form expressions, allowing efficient implementations of our self-shadowing model.

4.3. Simplified model with isotropic self-shadowing

Our model greatly simplifies when the self-shadowing function is isotropic, *i.e.* when $A(\omega) = A$, $\forall \omega$. As discussed in Sec. 6, this simplified model is easier to implement, requires less parameters at rendering and has sufficient accuracy in practice. In the case of isotropic self-shadowing, expressions reduce to

$$\sigma_t(\omega) = A \rho \int D(\omega_m) \langle \omega \cdot \omega_m \rangle d\omega_m \quad (4.8)$$

$$\sigma_{ss}(\omega) = \alpha_{ss}(\lambda) A \sigma_t(\omega) \quad (4.9)$$

$$f_{ss}(\omega \rightarrow \omega') = \frac{\rho \alpha_{ss}(\lambda) A^2 D(\omega_h)}{4 \sigma_{ss}(\omega)} \quad (4.10)$$

$$\sigma_{ms}(\omega) = \alpha_{ms}(\lambda) \sigma_t(\omega) (1 - A) \quad (4.11)$$

$$f_{ms}(\omega') = \frac{\sigma_{ms}(\omega')}{\int \sigma_{ms}(\omega'') d\omega''} = \frac{\sigma_t(\omega')}{\int \sigma_t(\omega'') d\omega''} \quad (4.12)$$

Note that $\sigma_t(\omega)$ is equal to A times the attenuation coefficient of the standard microflake model, and f_{ss} is exactly the specular phase function of the standard microflake model.

5. Implementing the self-shadowing model using trigonometric lobes \cos^{2n} and \sin^{2n}

As we highlighted before, functions in our shadowing model (Eq. 4.2, 4.3 and 4.7) involve spherical convolutions and integrals of microflake distributions D and self-shadowing functions A . The microflake normal distribution D is defined in the space of microflake normals, while A is defined in the space of incoming or outgoing directions. Because of this, it is very difficult to find closed-form expressions of integrals such as Eq. 4.3. For instance, we could not find solutions using the commonly used SGGX distribution, the Gaussian Fiber Distribution [ZJMB11], Spherical Gaussians [TS06] or Anisotropic Spherical Gaussians [XSD*13] despite their interesting mathematical properties.

Fortunately, we found closed-form expressions using microflake distributions based on trigonometric lobes $D(\omega) = \cos^{2n}(\omega, \xi_D)$ and $D(\omega) = \sin^{2n}(\omega, \xi_D)$ (Sec. 5.2), and using an appropriate self-shadowing function (Sec. 5.1). We give here our main results (expressions for σ_t and σ_{ss}). Complete derivations, proofs and sampling procedures can be found in our supplemental material.

5.1. Choosing the self-shadowing function A .

We introduce an anisotropic self-shadowing function A of the form

$$A(\omega) = \omega^T S \omega \quad (5.1)$$

with S a symmetric positive definite matrix encoding anisotropy as in SGGX distributions [HDCD15] (this function can be seen as the *square* projected area of an ellipsoid). As we want $0 < A(\omega) \leq 1$, we ensure that S has eigenvalues equal to or below 1. We found that this representation is simple and flexible enough for encoding smooth directional changes of the self-shadowing probability, and it can be stored with only 6 parameters.

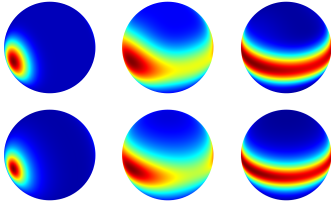


Figure 3: (top): trigonometric lobes (Eq. 5.2). (bottom): SGGX distributions. These distributions can represent similar foliage-like (left) and fiber-like (right) media.

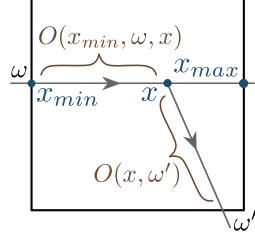


Figure 4: Notations for Eq. 7.10, for a ray in a block of input voxels or in a low-res voxel.

5.2. Using trigonometric lobes for distribution D .

Using self-shadowing functions described by Eq. 5.1, we found closed-form expressions for our model using the following microflake normal distributions:

$$D_{\cos}(\omega, \xi, n) = \frac{\cos^{2n}(\omega, \xi)}{N_{\cos}(n)} = \frac{(\omega \cdot \xi)^{2n}}{N_{\cos}(n)}$$

$$D_{\sin}(\omega, \xi, n) = \frac{\sin^{2n}(\omega, \xi)}{N_{\sin}(n)} = \frac{(1 - (\omega \cdot \xi)^2)^n}{N_{\sin}(n)}$$

with $n \in \mathbb{N}$ and $N_{\cos}(n)$ and $N_{\sin}(n)$ the normalization factors whose closed-form expressions can be found in our supplemental material. These distributions will be referred to as *trigonometric lobes* in this paper. Any linear combination of such lobes still leads to closed-form expressions in our model. These lobes can represent the same kinds of distributions as the SGGX distribution (Fig. 3), using for instance

$$D(\omega) = w_c D_{\cos}(\omega, \xi_c, n_c) + w_s D_{\sin}(\omega, \xi_s, n_s) + \frac{w_{iso}}{4\pi} \quad (5.2)$$

with w_c, w_s, w_{iso} positive weights such that $w_c + w_s + w_{iso} = 1$.

5.3. Attenuation coefficients for \cos^{2n} and \sin^{2n} lobes.

In our supplemental material, we derive a general closed-form expression for the attenuation coefficient σ_r^{\cos} , but this expression involves costly Gauss hypergeometric functions ${}_2F_1$ and cannot be evaluated efficiently at rendering. However, for each particular value n , we have found efficient closed-form expressions of the form:

$$\sigma_r^{\cos}(\omega) = \rho A(\omega) \begin{bmatrix} 1 \\ (\omega \cdot \xi_D)^2 \\ \dots \\ (\omega \cdot \xi_D)^{2n} \end{bmatrix} \cdot C_1 \quad (5.3)$$

with C_1 a $n+1$ vector of coefficients. We computed coefficients for each n between 1 and 20, using symbolic integration [MGH*05]. Note that for each n , these expressions are exact. We have found similar expressions for \sin^{2n} lobes (only coefficients C_1 change).

5.4. Single Scattering coefficients

Similarly, we have found closed-form expressions for σ_{ss} for each n :

$$\sigma_{ss}^{\cos}(\omega) = \alpha_{ss}(\lambda) \rho A(\omega) \begin{bmatrix} 1 \\ (\omega \cdot \xi_D)^2 \\ \dots \\ (\omega \cdot \xi_D)^{2n} \end{bmatrix} \cdot C_2 \cdot \begin{bmatrix} \omega^T S_A \omega \\ \xi_D^T S_A \xi_D \\ \text{trace}(A) \end{bmatrix} \\ + \alpha_{ss}(\lambda) \rho A(\omega) \begin{bmatrix} \omega \cdot \xi_D \\ \dots \\ (\omega \cdot \xi_D)^{2n-1} \end{bmatrix} \cdot C_3 \quad (5.4)$$

with C_2 a $(n+1) \times 3$ matrix of coefficients and C_3 a vector of coefficients of length n . We found similar expression for \sin^{2n} lobes.

5.5. Evaluating and sampling f_{ss} and f_{ms}

Details concerning phase functions can be found in our supplemental material. The evaluation of our multiple scattering phase function f_{ms} (Eq. 4.7) involves the integral of the multiple scattering coefficient:

$$\int \sigma_{ms}(\omega) d\omega = \alpha_{ms}(\lambda) \int \sigma_r(\omega) d\omega - \frac{\alpha_{ms}(\lambda)}{\alpha_{ss}(\lambda)} \int \sigma_{ss}(\omega) d\omega.$$

We provide closed-form expressions for $\int \sigma_r(\omega)$ and $\int \sigma_{ss}(\omega)$. We also provide sampling procedures for the visible normal distributions [HDCD15] of trigonometric lobes. We rely on rejection sampling for sampling exactly f_{ss} and f_{ms} . As discussed in our supplemental material, our sampling procedures are very efficient (*i.e.* samples are almost always accepted) when the shadowing function A has few anisotropic variations (which is often the case in practice) and are less efficient when A is highly anisotropic.

5.6. Limitations

Compared to SGGX distributions, implementing our model with the distribution proposed in Eq. 5.2 is about 2 to 3 times more expensive for evaluating $\sigma_r(\omega)$, and about 25 to 60 times more expensive for sampling the phase functions, depending on n_c, n_s , and $A(\omega)$. Despite these additional costs compared to the standard microflake model, rendering our LoDs is much faster than rendering high-resolution volumes. In the next section, we propose an implementation of our simplified self-shadowing model that is more efficient because it relies on SGGX for σ_r and f_{ss} .

Another limitation of the implementation proposed here is that unlike SGGX distributions, interpolating trigonometric distributions cannot be done by linear interpolation of trigonometric lobe parameters. This means that tri-linear spatial interpolation at rendering would require 8 evaluations of σ_r .

6. Implementing the simplified self-shadowing model using the SGGX distribution

In our simplified self-shadowing model, σ_r and σ_{ss} almost reduce to the standard microflake model, for which the SGGX distribu-

tion allows for efficient rendering. For this reason, we propose an implementation of the simplified model based on the SGGX distribution. However, using SGGX distributions, there is no closed-form expression for $\int \sigma_{ms}(\omega) d\omega$, which is involved in the multiple scattering phase function. Assuming normalized SGGX matrices, meaning that the highest eigenvalue is 1, the other two eigenvalues belong to the interval $(0, 1]$. We discretized this interval and pre-computed $\int \sigma_r(\omega'') d\omega''$ for each pair of eigenvalues using numerical integration.

In the simplified self-shadowing model, sampling the multiple scattering phase function f_{ms} is equivalent to importance sampling σ_r , i.e. the projected area of the SGGX ellipsoid given by $\sqrt{\omega^T S \omega}$. In the supplemental material, we provide the details of a sampling procedure for f_{ms} , using the fact that $\omega^T S \omega$ is close enough to $\sqrt{\omega^T S \omega}$ to allow efficient rejection sampling.

7. Downsampling with the microscopic self-shadowing model

In this section, we introduce new algorithms for downsampling standard microflake volumes using our microscopic self-shadowing model. We assume that input voxels contain one density value, parameters for the microflake distribution and a specular albedo. We describe two algorithms that we used for generating results in Sec. 8: **Aniso** uses anisotropic self-shadowing functions in each low-res voxel, and **Iso** relies on our simplified self-shadowing model (Sec. 4.3) for which each low-res voxel needs less parameters. We also describe in this section two naïve algorithms that use the standard microflake model in output voxels: **Linear** performs linear pre-filtering for densities and albedos, and **Transp** performs linear pre-filtering for albedos but computes density parameters that preserve local transparency. These naïve algorithms are used for comparison in Sec. 8.

Fig. 5 illustrates our downsampling pipeline and summarizes our input and output models for each algorithm. Each of our output voxels have a unique single scattering albedo, which could be insufficient for downsampling datasets with strong correlations between microflake distributions and albedos (e.g. multi-yarn fabrics). We discuss this case in Sec. 8.4.

7.1. Overview

In the four algorithms described in this section, each block of input voxels (Fig. 5) is downsampled into one low-res voxel independently of other low-res voxels. Parameters of each low-res voxel are computed using different strategies summarized in the following table:

| | ρ | α_{ss} | A | α_{ms} |
|---------------|------------------|------------------|-------------------|---------------|
| Linear | Linear (Eq. 7.1) | Linear (Eq. 7.2) | - | - |
| Transp | Mean (Eq. 7.5) | Linear (Eq. 7.2) | - | - |
| Aniso | Max (Eq. 7.7) | Linear (Eq. 7.2) | Sec. 7.4.2, 7.4.1 | Sec. 7.4.4 |
| Iso | Mean (Eq. 7.6) | Linear (Eq. 7.2) | Sec. 7.4.2 | Sec. 7.4.4 |

We do not contribute to microflake pre-filtering. Algorithms described in this section are valid for any number of SGGX or

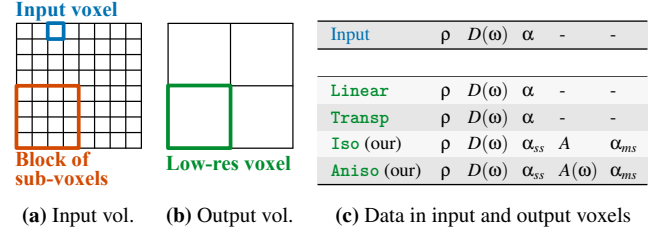


Figure 5: Our input voxels (a) have parameters for the standard microflakes model (density ρ , microflake distribution $D(\omega)$ and albedo α). Each low-resolution voxel (b) approximates a cube of input voxels (or block of input voxels). In this paper, we compare four downsampling algorithms (c). **Linear** and **Transp** use the standard microflake model. **Iso** and **Aniso** use our self-shadowing model and output voxels store additional parameters in this case, for the self-shadowing function A and for the multiple scattering albedo α_{ms} .

trigonometric lobes in input and output voxels. In our implementation, we used a single SGGX lobe in each low-resolution voxel, or its equivalent using trigonometric lobes (Fig. 3), and we pre-filtered normal distributions using linear SGGX pre-filtering [HDCD15].

7.2. Algorithm Linear

This is the naïve linear algorithm: given ρ_i and α_i , densities and albedos of N_{iv} input voxels (Fig. 5), the parameters of the corresponding low-resolution voxel are computed using

$$\rho = \frac{1}{N_{iv}} \sum \rho_i \quad (7.1)$$

and

$$\alpha = \frac{\sum \alpha_i \rho_i}{\sum \rho_i}. \quad (7.2)$$

7.3. Algorithm Transp

This algorithm first computes transparency of the block of input voxel in the three axis of the voxel grid: $T(\omega_X)$, $T(\omega_Y)$ and $T(\omega_Z)$. These quantities are computed exactly from directional transparencies of each input voxel. Given the width of input voxels L_{iv} , the transparency of one input voxel in direction ω_i is given by

$$\exp\left(-L_{iv} \rho_i \int D_i(m) \langle \omega_i \cdot m \rangle dm\right). \quad (7.3)$$

Similarly, transparency of the low-resolution voxel in direction ω_i is given by

$$\exp(-L \sigma_r(\omega_i)) = \exp\left(-L \rho \int D(m) \langle \omega_i \cdot m \rangle dm\right) \quad (7.4)$$

with L the size of the low-resolution voxel. We compute density parameters that preserve transparency in each direction ω_i , and we average them:

$$\rho = \frac{1}{3} \sum_{i \in \{X, Y, Z\}} \frac{-\log(T(\omega_i))}{L \int D(m) \langle \omega_i \cdot m \rangle dm}. \quad (7.5)$$

7.4. Downsampling using our self-shadowing model (Iso and Aniso)

Given a pre-filtered microflake distribution D , our algorithms first compute directional transparencies of the block of input voxels as in algorithm **Transp**. Then, they compute a temporary density value ρ_0 and temporary parameters for a self-shadowing function A_0 . These temporary parameters are used for estimating the amount of self-shadowing in the low-resolution voxel. The amount of self-shadowing is also computed in the block of input voxels (Fig. 5). The final density ρ and self-shadowing parameters are computed so that the amount of self-shadowing is the same in the block of input voxels and the low-res voxel. Single scattering albedos are downsampled linearly using Equation 7.2, and the computation the multiple scattering albedo is discussed in Sec. 7.4.4.

7.4.1. Computing temporary parameters ρ_0 and A_0 from directional transparencies

Given the size of low-res voxels L , the transparency of a low-res voxel in direction ω_i is given by

$$\exp(-L\sigma_r(\omega_i)) = \exp\left(-L\rho_0 A_0(\omega_i) \int D(m)\langle\omega_i \cdot m\rangle dm\right).$$

Algorithm **Iso** only uses the simplified microflake model in which $A_0(\omega) = A_0, \forall\omega$. In this case, we set $A_0 = 1$ and compute ρ for preserving the average transparency as in algorithm **Transp**:

$$\rho_0 = \frac{1}{3} \sum \frac{-\log(T(\omega_i))}{L \int D(m)\langle\omega_i \cdot m\rangle dm}. \quad (7.6)$$

Algorithm **Aniso** uses the anisotropic self-shadowing function introduced in Section 5.1. It computes ρ_0 using

$$\rho_0 = \max_{i \in \{X,Y,Z\}} \left(\frac{-\log(T(\omega_i))}{L \int D(m)\langle\omega_i \cdot m\rangle dm} \right), \quad (7.7)$$

and then it computes the self-shadowing parameters with

$$A_0(\omega) = \omega^T \begin{bmatrix} s_X & 0 & 0 \\ 0 & s_Y & 0 \\ 0 & 0 & s_Z \end{bmatrix} \omega \quad (7.8)$$

with

$$s_i = \frac{-\log(T(\omega_i))}{\rho_0 L \int D(m)\langle\omega_i \cdot m\rangle dm}, \quad i \in \{X,Y,Z\} \quad (7.9)$$

so that directional transparencies of the low-res voxel matches exactly $T(\omega_x)$, $T(\omega_y)$ and $T(\omega_z)$, the transparencies of the block of input voxels.

7.4.2. Estimating self-shadowing

At this stage, the low-resolution voxel has correct transparency but incorrect self-shadowing probability. For preserving self-shadowing, we estimate the amount of single scattering in the block of input voxels and in the low-res voxel, and we compute final parameters accordingly. More precisely, we estimate the mean amount of energy that leaves the block of input voxels after exactly one scattering event. For one incoming ray r with direction ω_r , intersecting the block of input voxels in x_{\min} and x_{\max} as shown in

Fig. 4, the amount of single scattering writes

$$P_{ss}(r) = \int_{x_{\min}}^{x_{\max}} \sigma_s(x, \omega_r) O(x_{\min}, \omega_r, x) \int_{S^2} f(x, \omega_r \rightarrow \omega) O(x, \omega) d\omega dx \quad (7.10)$$

with

$$O(x_1, \omega, x_2) = \exp\left(-\int_{x_1}^{x_2} \sigma_t(s, \omega) ds\right) \quad (7.11)$$

and $O(x_1, \omega)$ a similar transmittance probability between x_1 and the limit of the block of input voxels, in direction ω (Fig. 4). The quantity $P_{ss}(r)$ is dimensionless but wavelength dependent. We estimate the mean amount of single scattering in the block of input voxels by casting N_{ray} rays all around the block of input voxels. In our implementation, we chose rays with directions parallel to the mains axis of the voxel grid. For each ray, we compute $P_{ss}(r_i)$ and we average the results:

$$M_{ss}^{\text{input}} = \frac{1}{N_{ray}} \sum_{\text{rays}} P_{ss}(r_i). \quad (7.12)$$

Similarly, we estimate the quantity M_{ss} , the amount of single scattering in the low-res voxel, using the pre-filtering distribution D , ρ_0 , $A_0(\omega)$ and the corresponding σ_{ss} and f_{ss} (instead of σ_s and f) in Eq. 7.10.

7.4.3. Computing final density and self-shadowing parameters

Thanks to our model, we can control the amount of self-shadowing in the low-res voxel while preserving the voxel transparency, using $\rho = \frac{\rho_0}{\gamma}$ and $A(\omega) = \gamma A_0(\omega)$ with $\gamma \in (0, 1]$. Indeed, for any value γ , using ρ and $A(\omega)$ instead of ρ_0 and $A_0(\omega)$ in the low-res voxel does not impact the attenuation coefficient:

$$\sigma_t(\omega) = \rho A(\omega) \int D(m)\langle\omega \cdot m\rangle dm = \rho_0 A_0(\omega) \int D(m)\langle\omega \cdot m\rangle dm.$$

On the contrary, the single scattering coefficient is multiplied by γ :

$$\begin{aligned} \sigma_{ss}(\omega) &= \rho A(\omega) \int A(\omega) D(m)\langle\omega \cdot m\rangle dm \\ &= \gamma \rho_0 A_0(\omega) \int A_0(\omega) D(m)\langle\omega \cdot m\rangle dm. \end{aligned}$$

This means that the amount of self-shadowing in the low-res voxel (estimated from Eq. 7.10) would now be γM_{ss} . Given M_{ss}^{input} and M_{ss} , we compute the value γ that minimizes the error over various wavelengths:

$$\gamma = \arg \max_g \sum_{\lambda_i} \left\| M_{ss}^{\text{input}}(\lambda_i) - g M_{ss}(\lambda_i) \right\|^2 \quad (7.13)$$

λ_i being wavelengths. We used RGB albedos in our implementation.

7.4.4. Computation of multiple scattering albedo σ_{ms}

Our self-shadowing model requires a multiple scattering albedo σ_{ms} that characterizes the color of light scattered at least two times at the *microscopic* scale due to self-shadowing. At rendering, multiple scattering occurs at the microscopic scale with our model, but

it can also occur several times in the low-res voxel. We need to estimate σ_{ms} such that the *effective* albedo α_e of the low-res voxel, that is the resulting albedo of the voxel taking into account multiple scattering with microscopic and non-microscopic scattering events, is the same as the effective albedo α_e^{input} of the corresponding block of input voxels.

We first estimate the effective albedo α_e^{input} in the block of input voxels, that is the average color of light when it leaves the block after 1 or more scattering events. We cast rays through the block of input voxels as for single scattering estimations (Fig. 5). For each ray, we compute a light path until the ray leaves the block, exactly as a volume path tracer would do. Then, α_e^{input} is computed averaging ray throughputs.

Now we want to find α_{ms} such that $\alpha_e = \alpha_e^{\text{input}}$. Unfortunately, there is no simple way to derive exactly α_e from α_{ms} , and we want to avoid iterative optimizations because they are time consuming. We approximate α_e using:

$$\alpha_e \approx \sum P_{sh}(n)(P_{ss}\alpha_{ss} + (1 - P_{ss})\alpha_{ms})^n \quad (7.14)$$

with $P_{sh}(n)$ the probability for a ray of having exactly n scattering events before leaving the voxel ($\sum P_{sh}(n) = 1$) and P_{ss} the probability of self-shadowing at the microscopic scale. We cast rays in the low-res voxel and estimate probabilities $P_{sh}(n)$ for each n (up to a given value), which do not depend on σ_{ms} . For P_{ss} , we use the following approximation:

$$P_{ss} \approx \frac{\int \sigma_{ss}}{\alpha_{ss}(\lambda) \int \sigma_t}. \quad (7.15)$$

Finally, we can estimate α_{ms} by solving numerically the equation

$$\alpha_e^{\text{input}} = \sum P_{sh}(n)(P_{ss}\alpha_{ss} + (1 - P_{ss})\alpha_{ms})^n \quad (7.16)$$

for each wavelength. We evaluated this method by computing the relative errors between the effective albedos of low-res voxels α_e , measured with ray casting, and the effective albedos α_e^{input} in the corresponding blocks of input voxels. Our method (Eq. 7.14 and 7.15) tends to slightly underestimate the effective albedo α_e , and its accuracy depends on the density and the amount of shadowing in the voxel. Details can be found in Sec. 6 of the supplemental material, including statistics of errors for the assets used in the next section.

8. Implementation and results

8.1. Pre-computation time

Pre-computation time depends on the number of samples used for estimating self-shadowing probabilities and multiple scattering albedos (Sec. 7.4.2 and 7.4.4). In our implementation, we adapted the number of samples to the complexity of the block of input voxels. For LoD i (*i.e.* 2^i times smaller than input volume in each dimension), blocks of input voxels viewed from one side have a complexity of 4^i voxels. We cast 40×4^i rays for each non-empty low-res voxels so that our LoDs are not subject to noise. We got the following pre-computation times on a Intel Xeon Processor E5-2630 v3 (in CPU core hour):

| | Bunny | hair | Cedar | Pine | Hairy ball |
|--------------------|----------------|---------|---------|---------|------------|
| Size of input vol. | $\simeq 500^3$ | 600^3 | 680^3 | 680^3 | 680^3 |
| Aniso / LoD 1 | 1.1 | 1.4 | 0.1 | 0.13 | 6.8 |
| Aniso / LoD 3 | 0.92 | 0.6 | 0.14 | 0.17 | 3.2 |
| Aniso / LoD 5 | 0.64 | 0.33 | 0.17 | 0.15 | 1.2 |

In their work, Zhao, Wu et al. give pre-computation time of respectively 12 and 40 CPU core hours for the bunny (LoD 2) and the hairy ball (LoD 3). We obtained accurate results for these model with respectively 0.97 and 3.2 CPU core hours. As our algorithm downsamples each block of input voxels independently, parallelizing our method is straightforward.

8.2. Savings

Compared to the standard microflake model, our model requires additional data per voxel: 2 additional values for our trigonometric distributions compared to SGGX, 3 for multiple scattering albedos (RGB) and 6 or 1 additional values for the shadowing function using respectively our anisotropic self-shadowing model or our simplified model. Despite its additional parameters, our downsampling method allows for huge memory savings as shown in the following table:

| | Linear | Iso (SGGX) | Aniso (triglobe) |
|------------------------|--------|------------|------------------|
| Per voxel values | 10 | 14 | 21 |
| Size of LoD1 wrt input | 12.5% | 17.5% | 26.3% |
| Size of LoD3 wrt input | 0.20% | 0.27% | 0.41% |

On average, for the LoDs shown in Fig. 6, the render time for naïve LoDs is around 11% of the render time for high-resolution volumes, and around 13% for our LoDs with algorithm **Aniso**. This means that our LoDs also decrease render time despite the additional cost compared to rendering naïve LoDs.

8.3. Results

Fig. 6 compares our LoDs with input volumes and results from naïve algorithms described in Section 7. Our downsampling method supports semi-transparent inputs with low-density voxels. Our LoDs are only slightly more accurate in this case because naïve methods already perform well when there are no strong self-shadowing effects (Fig. 6a and 6f). However, our LoDs are much more accurate than naïve LoDs when input volumes have relatively dense voxels (Fig. 6b to 6e). Because we preserve local transparency, our results have correct silhouettes even when input datasets have intricate shapes (Fig. 6d and 6e). Our LoDs can be computed at arbitrary scales and their appearance is consistent (Fig. 1).

Using anisotropic self-shadowing functions allows for preserving exactly directional transparency in low-res voxels independently of the microflake distribution, for instance when the microflake distribution is isotropic as shown in Fig. 2d. However, we found in our experiments that using anisotropic self-shadowing does not improve accuracy significantly in practice,

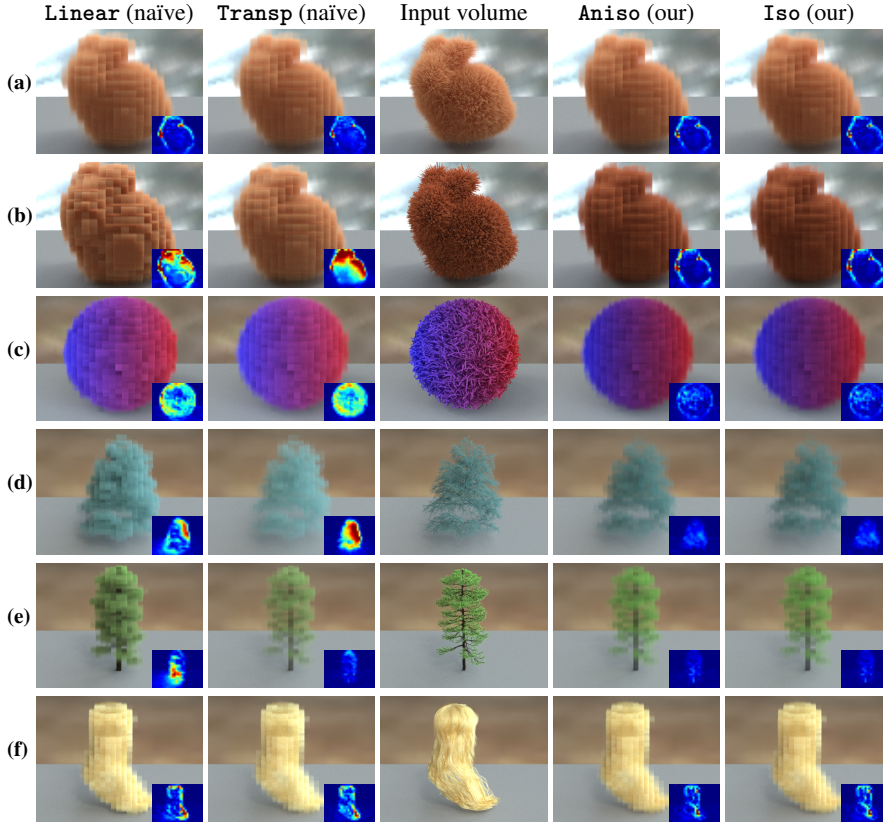


Figure 6: Comparison of volumes downsampled using the four algorithms described in Sec. 7. Low-res volumes are LoD 5, meaning that the number of voxel has been divided by 8^3 compared to input volumes. Insets show relative L_1 errors in linear RGB, computed on low-resolution pictures. (a): low-density bunny. (b): high density bunny ($\times 20$ compared to (a)). (c): dense hairy ball with anisotropic microflake distributions. (d): cedar foliage with homogeneous albedo. (e): pine tree with dark trunk. (f): hair with anisotropic microflake distribution. Our algorithm increases significantly accuracy of LoDs when input volumes have dense voxels ((b) to (e)).

even for datasets with dense aligned voxels such as the hairy bunny (Fig. 6b). This means that our simplified model has sufficient accuracy in most downsampling applications, and that accuracy is more limited by albedos and phase functions than by accurate view-dependent attenuation.

We compared our LoDs with results provided by Zhao, Wu et al. [ZWDR16]. Fig. 7 and Fig. 8 show LoDs of the dense hairy bunny and the velvet datasets. Both methods preserve large-scale albedos, unlike naïve methods (1, 6b). Silhouettes of our LoDs are more accurate because we preserve local transparency in low-resolution voxels instead of pre-filtering density linearly.

8.4. Limitations

The main limitation of our work is that the self-shadowing functions we use do not model accurately self-shadowing effects in dense voxels. Indeed, our functions are symmetric ($A(\omega) = A(-\omega)$) while self-shadowing in a dense voxel is very strong forward and

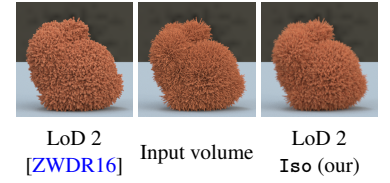


Figure 7: Dense hairy bunny. Both methods preserve the macroscopic appearance of the input volume. Silhouettes are slightly more accurate in our LoD because we do not pre-filter density linearly.

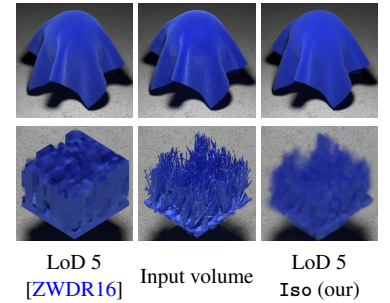


Figure 8: Close-up vs large scale view of the velvet dataset. Our LoDs have accurate transparency (bottom right). Because they have dense low-resolution voxels (top left), LoDs from [ZWDR16] preserve more accurately fabrics-like appearance at large scales, especially at grazing angles. Preserving both accurate transparency and scattering behavior remains an open problem for such dense input volumes.

much smaller backward. Moreover, our multiple scattering phase function is relatively diffuse, while light scattered multiple times in a dense voxel mainly leaves this voxel backward – because light cannot reach the opposite side of the voxel. Hence, our phase functions lack accuracy when input datasets are very dense and heterogeneous. This can be seen in Figure 8 where the LoD from [ZWDR16] better preserves velvet-like reflections at grazing angles because they use dense low-resolution voxels. Fig. 9 shows that our LoDs lack accuracy for the hair datasets because input voxels are too dense. Our self-shadowing model allows for preserving the mean amount of local self-shadowing, but preserving both local transparency and accurate scattering behavior remains an open problem for dense and heterogeneous input volumes.

We did not address the problem of colored multi-fiber datasets, for which it is important to store multiple albedos and lobes in low-resolution voxels. We believe that our method can be extended to such case using self-shadowing estimations for each lobe, but a rigorous study remains to be done. We did not work on animated

datasets with time varying scattering parameters in input voxels. Pre-computing low-resolution volumes at each time step would reduce the benefits of using LoDs in some applications.

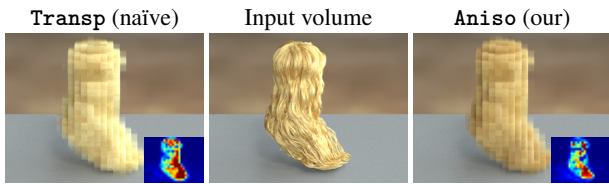


Figure 9: Same hair dataset as in Fig. 6f, with density multiplied by 10. This is a challenging case because of very anisotropic microflake distributions and complex self-shadowing effects due to dense input voxels. Our LoDs are slightly better than naïve methods for this datasets but lacks accuracy (loss of bright anisotropic reflections). This is because our self-shadowing model does not preserve well appearance of very dense voxels (Sec. 8.4).

9. Summary

We have introduced a new participating medium model based on the microflake model, with new parameters characterizing self-shadowing effects at a microscopic scale. This model allows for controlling independently the directional transparency of a medium and the amount of multiple scattering that occurs at the microscopic scale. Based on this model, we have introduced a new volume downsampling approach that preserves both transparency and self-shadowing effects in low-resolution volumes.

Our downsampling approach overcomes several limitations of previous work: it does not produce artifacts on spatially varying data, and it can handle semi-transparent datasets or intricate shapes with hidden parts. Our method also reduces pre-computation time because computing local self-shadowing estimations is faster than large scale training rendering. We tested our algorithm on several challenging datasets and have shown that our method can down-sample dense heterogeneous volumes for which naïve methods are inaccurate, as well as datasets with complex silhouettes such as trees. Our LoDs have consistent appearance through scales, allowing for huge storage savings and drastic reduction of rendering time with little loss of quality.

Our self-shadowing model can be used for adjusting anisotropic attenuation in low-resolution voxels. However, from our results, we have drawn the conclusion that accuracy in LoDs does not depend much on accurate anisotropic transparency: our simplified self-shadowing model has proven to be as accurate as our full model for all the datasets we tested. To the contrary, we observed that accuracy of low-res volumes highly depends on estimated albedos and phase functions. We found that preserving both local transparency and accurate phase functions remains an open problem in the case of input volumes with very dense voxels. In our model, shadowing parameters mainly control the amount of self-shadowing, but our self-shadowing functions do not model accurately what happens in dense voxels. Future work may focus on new self-shadowing model for increasing accuracy of phase functions in such cases.

References

- [Cha50] CHANDRASEKAR S.: *Radiative Transfer*. Oxford Univ. Press, 1950. 2
- [CNS*11] CRASSIN C., NEYRET F., SAINZ M., GREEN S., EISEMANN E.: Interactive indirect illumination using voxel cone tracing. *Computer Graphics Forum* 30, 7 (2011), 1921–1930. 3
- [HDCD15] HEITZ E., DUPUY J., CRASSIN C., DACHSBACHER C.: The SGGX microflake distribution. *ACM Trans. Graph.* 34, 4 (July 2015), 48:1–48:11. 2, 3, 4, 5, 6, 7
- [Jak10] JAKOB W.: Mitsuba renderer, 2010. 3
- [JAM*10] JAKOB W., ARBREE A., MOON J. T., BALA K., MARSCHNER S.: A radiative transfer framework for rendering materials with anisotropic structure. *ACM Trans. Graph.* 29, 4 (July 2010), 53:1–53:13. 2, 3, 4
- [KB08] KRAUS M., BÜRGER K.: Interpolating and downsampling rgba volume data. In *13th International Fall Workshop Vision, Modeling, and Visualization 2008* (Jan. 2008), pp. 323–332. 3
- [KSZ*15] KHUNGURN P., SCHROEDER D., ZHAO S., BALA K., MARSCHNER S.: Matching real fabrics with micro-appearance models. *ACM Trans. Graph.* 35, 1 (Dec. 2015), 1:1–1:26. 2
- [LN17] LOUBET G., NEYRET F.: Hybrid mesh-volume LoDs for all-scale pre-filtering of complex 3D assets. *Comput. Graph. Forum* 36, 2 (May 2017), 431–442. 2
- [MGH*05] MONAGAN M. B., GEDDES K. O., HEAL K. M., LABAHN G., VORKOETTER S. M., MCCARRON J., DEMARCO P.: *Maple 10 Programming Guide*. Maplesoft, Waterloo ON, Canada, 2005. 6
- [MPG*16] MÜLLER T., PAPAS M., GROSS M., JAROSZ W., NOVÁK J.: Efficient rendering of heterogeneous polydisperse granular media. *ACM Trans. Graph.* (2016), 168:1–168:14. 2
- [MPH*15] MENG J., PAPAS M., HABEL R., DACHSBACHER C., MARSCHNER S., GROSS M., JAROSZ W.: Multi-scale modeling and rendering of granular materials. *ACM Trans. Graph.* 34, 4 (July 2015), 49:1–49:13. 2
- [MWM07] MOON J. T., WALTER B., MARSCHNER S. R.: Rendering discrete random media using precomputed scattering solutions. *EGSR'07*, pp. 231–242. 2
- [MWM08] MOON J. T., WALTER B., MARSCHNER S.: Efficient multiple scattering in hair using spherical harmonics. *ACM Trans. Graph.* 27, 3 (Aug. 2008), 31:1–31:7. 2
- [Ney98] NEYRET F.: Modeling, animating, and rendering complex scenes using volumetric textures. *IEEE Transactions on Visualization and Computer Graphics* 4, 1 (Jan. 1998), 55–70. 2
- [SKZ11] SCHRÖDER K., KLEIN R., ZINKE A.: A volumetric approach to predictive rendering of fabrics. *EGSR '11*, pp. 1277–1286. 2
- [TS06] TSAI Y.-T., SHIH Z.-C.: All-frequency precomputed radiance transfer using spherical radial basis functions and clustered tensor approximation. In *Proceedings of SIGGRAPH'06* (2006), pp. 967–976. 5
- [WMLT07] WALTER B., MARSCHNER S. R., LI H., TORRANCE K. E.: Microfacet models for refraction through rough surfaces. In *Proceedings of EGSR'07* (2007), pp. 195–206. 4
- [XSD*13] XU K., SUN W.-L., DONG Z., ZHAO D.-Y., WU R.-D., HU S.-M.: Anisotropic spherical gaussians. *ACM Trans. Graph.* 32, 6 (Nov. 2013), 209:1–209:11. 5
- [ZJMB11] ZHAO S., JAKOB W., MARSCHNER S., BALA K.: Building volumetric appearance models of fabric using micro ct imaging. In *Proceedings of SIGGRAPH '11* (2011), pp. 44:1–44:10. 2, 3, 5
- [ZWDR16] ZHAO S., WU L., DURAND F., RAMAMOORTHY R.: Downsampling scattering parameters for rendering anisotropic media. *ACM Trans. Graph.* 35, 6 (2016). 1, 2, 3, 10