Fast Rendering of Fabric Micro-Appearance Models Under Directional and Spherical Gaussian Lights

PRAMOOK KHUNGURN, Cornell University
RUNDONG WU, Cornell University
JAMES NOECKEL, Cornell University
STEVE MARSCHNER, Cornell University
KAVITA BALA, Cornell University

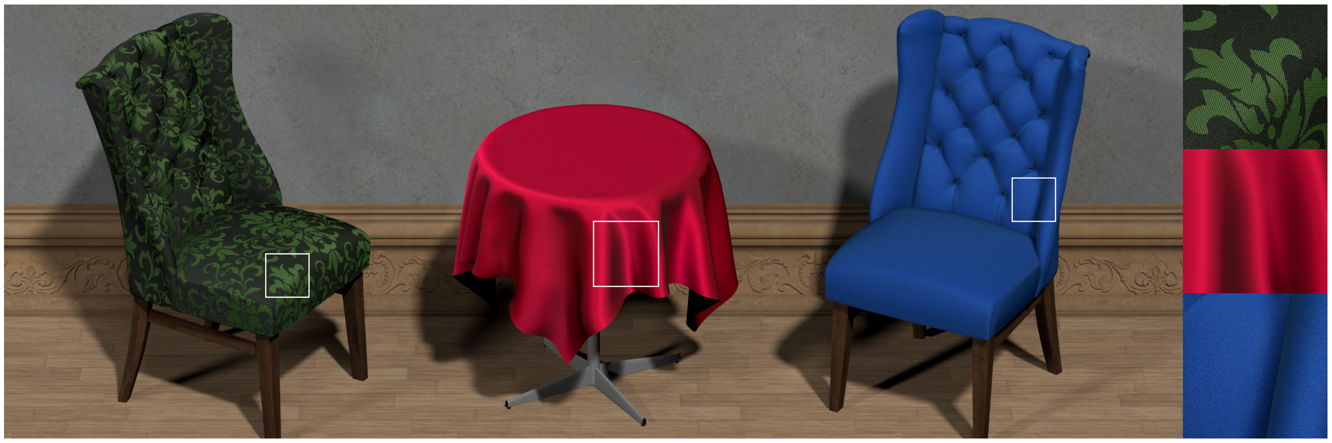


Fig. 1. Our algorithm enables fast, high-quality rendering of fabric models made up of individual fibers with multiple scattering. The table was rendered in 63 seconds, the green chair 291 seconds, and the blue chair 131 seconds on a single Nvidia GeForce GTX 970 GPU.

Rendering fabrics using micro-appearance models—fiber-level microgeometry coupled with a fiber scattering model—can take hours per frame. We present a fast, precomputation-based algorithm for rendering both single and multiple scattering in fabrics with repeating structure illuminated by directional and spherical Gaussian lights.

Precomputed light transport (PRT) is well established but challenging to apply directly to cloth. This paper shows how to decompose the problem and pick the right approximations to achieve very high accuracy, with significant performance gains over path tracing. We treat single and multiple scattering separately and approximate *local* multiple scattering using precomputed transfer functions represented in spherical harmonics. We handle shadowing between fibers with precomputed per-fiber-segment visibility functions, using two different representations to separately deal with low and high frequency spherical Gaussian lights.

Our algorithm is designed for GPU performance and high visual quality. Compared to existing PRT methods, it is more accurate. In tens of seconds on a commodity GPU, it renders high-quality supersampled images that take path tracing tens of minutes on a compute cluster.

0730-0301/2017/11-ART232 $15.00
https://doi.org/10.1145/3130800.3130829

CCS Concepts: • **Computing methodologies → Rendering**;

Additional Key Words and Phrases: cloth, precomputed radiance transfer, rendering

ACM Reference format:
Pramook Khungurn, Rundong Wu, James Noeckel, Steve Marschner, and Kavita Bala. 2017. Fast Rendering of Fabric Micro-Appearance Models Under Directional and Spherical Gaussian Lights. *ACM Trans. Graph.* 36, 6, Article 232 (November 2017), 15 pages.
https://doi.org/10.1145/3130800.3130829

1 INTRODUCTION

Fabrics are critical in many graphics applications including games, movies, and virtual prototyping. Recent fabric appearance models have achieved high fidelity by coupling fiber-level light scattering models with fiber geometry acquired from micron-resolution CT images of real fabric samples [Khungurn et al. 2015; Zhao et al. 2011]. However, rendering these *micro-appearance models* using path tracing is prohibitively slow because tracing rays through the complex microgeometry is costly, and long paths must be evaluated to capture multiple scattering inside the material. The process can take tens of core-hours per frame. Using precomputation to bypass these long paths is thus a logical choice to speed up rendering.

This paper presents a fast, precomputation-based algorithm for approximately rendering fabric micro-appearance models under directional and/or spherical Gaussian lights, which can be used to approximate environment illumination. Because our algorithm traces only eye rays through microgeometry, it is very suitable for GPU implementation. In particular, our implementation is able to render high-resolution, supersampled images of micron-resolution fabrics in tens of seconds, using only a single commodity GPU.

Our approach is enabled by focusing on fabrics with regular structure and limiting our attention to the effects of multiple scattering *inside* the fabric volume, in contrast to global interreflections.

Even with these assumptions, the problem is very challenging because one has to not only take into account complex occlusion among highly specular cloth fibers but also handle spherical Gaussian lights of all frequencies. We found that existing precomputed radiance transfer (PRT) techniques could not produce all aspects of fabric appearance—especially the specular reflection—and so had to search for a working alternative. We discovered that the problem can be effectively solved by decomposing it into several parts. Our contribution is identifying these parts and the right precomputation to use for each.

In particular, we first decompose scattered radiance into single and multiple scattering components. Because the latter varies rather smoothly over the fabric volume even when lighting is sharp, we approximate it with precomputed transfer functions based on low-order spherical harmonics.

Approximating single scattering under spherical Gaussian lights of all frequencies is more difficult. We classify spherical Gaussian lights into "sharp" and "soft" and use a different representation for precomputed visibility in each case. We use the spherical signed distance function (SSDF) for sharp spherical Gaussian lights, while for soft lights we approximate visibility as a sum of spherical Gaussians arranged over the sphere. Lastly, we tackle the highly specular reflectance with an analytic approximation to the convolution between a sharp spherical Gaussian light and the scattering function of a fiber, and we show how to use this approximation with the visibility representations to approximate the full shading integral.

We demonstrate high fidelity results, individually rendered in under a minute, for fabrics ranging from regularly structured woven textiles to a knitted fleece with unorganized surface structure. Our approach makes micro-appearance models practical for applications that require quick turnaround, such as interactive textile design.

2 BACKGROUND AND PREVIOUS WORK

Fabric appearance modeling. A piece of fabric can be modeled with a surface that captures its overall shape [Adabala et al. 2003; Irawan and Marschner 2012; Sadeghi et al. 2013; Sattler et al. 2003]. These models, however, abstract away microgeometry and account for multiple scattering in ad hoc ways.

Our work is based on the micro-appearance approach, in which fibers are explicitly represented, either by volumetric data [Zhao et al. 2011] or meshes [Schröder et al. 2014]. Appearance can be modeled by a phase function [Zhao et al. 2011] or a fiber-based light scattering model (also called a BCSDF) [Marschner et al. 2003; Zinke and Weber 2007]. Microgeometry makes it possible to see extreme details in close-ups but makes rendering much harder.

Our modeling process (Figure 2) starts with an *exemplar*: a small, rectangular piece of fabric modeled with explicit fibers. The exemplar is divided into *blocks*, and larger fabrics are synthesized by tiling

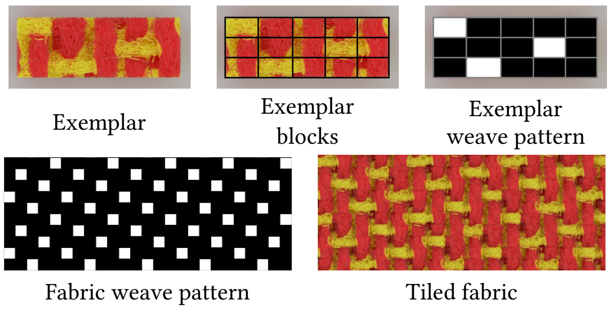
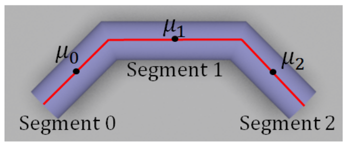


Fig. 2. Micro-appearance fabric model.

instances of these blocks according to weave patterns, then deforming them into arbitrary shapes using shell mapping [Porumbescu et al. 2005; Zhao et al. 2012].

We do precomputation and CPU rendering with a fiber mesh representation, in contrast to the volumetric representation of Zhao et al. [2011]; however, when using a GPU, we rasterize the fibers into a volume to better match the strengths of that architecture.



A *fiber* is represented by a 3D polyline of multiple straight *segments*. The kth segment's midpoint is denoted by μ_k. Light scattering from fibers is modeled by the two-term BCSDF by Khungurn et al. [2015].

Accelerated rendering of fiber assemblies and other discrete random media. A fiber assembly is challenging to render because its overall color arises from multiple scattering. Two-pass algorithms [Moon and Marschner 2006; Moon et al. 2008] compute single and multiple scattering separately to significantly accelerate the latter. We employ this separation and accelerate both components. Modular flux transfer (MFT) [Zhao et al. 2013] is a two-pass algorithm designed to render fabrics with repeating units. It is, however, not GPU-friendy because it employs photon tracing and recursive eye-ray tracing for the first 2 to 6 bounces.

Dual scattering [Zinke et al. 2008] significantly accelerates multiple scattering computation in a hair assembly. Nevertheless, we discovered that it is not suitable for our application. Ren et al. [2010] extends dual scattering to enable interactive rendering under spherical Gaussian lights, and Xu et al. [2011] enables interactive editing of BCSDF parameters. Both works approximate the convolution between the BCSDF and the spherical Gaussian light analytically, which we follow. However, they only coarsely approximate occlusion by the fiber volume, while we handle detailed occlusion by nearby fibers.

Iwasaki et al. [2014] presented an interactive algorithm for rendering fabrics modeled by the microcylinder model of Sadeghi et al. [2013] under environment lights. However, since it fundamentally uses a BRDF model, it offers no geometric details.

A fiber assembly is an example of a discrete random medium in which the bulk of the material consists of a large number of tiny discrete objects. Moon et al. [2007] describes a general technique for rendering such materials, and Meng et al. [2015] and Müller et al. [2016] specialize to those consisting of small grains. While these works deal with simulating full light transport, we focus on fast rendering and take into account only local multiple scattering, which is sufficient to produce realistic fabric colors.

Precomputed Radiance Transfer. In PRT, a transfer function from environment lighting to incoming light at a shaded point is represented as a transformation in a basis, whose choice has been extensively explored [Lehtinen and Kautz 2003; Ng et al. 2004; Sloan et al. 2002; Tsai and Shih 2006; Wang et al. 2009, 2006]. Because the whole collection of transfer functions is often too large to reside in GPU memory, researchers have proposed compression algorithms [Sloan et al. 2003; Tsai and Shih 2006]. For a more complete survey of PRT techniques, we refer the reader to the paper by Ramamoorthi [2009]. More recent works give transfer functions for scene elements that can be recombined at runtime [Loos et al. 2011], and methods to summarize scattering inside detailed scene objects, which are used to accelerate path tracing [Blumer et al. 2016].

Our algorithm differs from standard PRT algorithms in two ways. First, our algorithm precomputes on a flat fabric that is then warped into the final geometry, instead of considering only static scene geometry, making it similar in its goals to local, deformable PRT [Sloan et al. 2005]. Second, we separate single and multiple scattering, while standard PRT algorithms either do not separate them or deal only with single scattering. We approximate single scattering with precomputed visibility functions (which will be discussed in a moment) and multiple scattering with transfer functions specialized to indirect illumination. The transfer functions are represented by a bidirectional spherical harmonics (SH) structure, similar to the interreflection transfer function proposed by Pan et al. [2007].

Visibility Decomposition. When computing single scattering, we decompose visibility into *local* and *global* visibility. Local visibility involves occlusion by nearby fiber microgeometry, and global visibility involves occlusion by faraway parts of the fabric or other objects in the scene. Only local visibility is precomputed, and we regard it as a function of one direction stored at each fiber segment.

Our decomposition is similar to the one used in the work of Schröder et al. [2011], which is primarily concerned with representing fabrics using coarse volumes. Their local visibility term is a bidirectional function called the *bidirectional visibility distribution function* (BVDF), which can be thought of as an average of visibility functions at individual fiber segments in the voxel. While the BVDF would be the same as our visibility function when the voxels become fine enough, the BVDF's representation in Schröder et al.'s paper only works well with lights that have delta distributions (i.e., directional lights or point lights). We, on the other hand, give representations for local visibility that also work with spherical Gaussian lights.

Notation for spherical harmonics. We denote a real spherical harmonic (SH) basis function with $Y_j(\omega)$ where $\omega \in S^2$, and j serves as a single index. SH functions are classified into orders with $2l + 1$ functions having order l. Thus, a representation using SH of order up to L has $(L + 1)^2$ coefficients.

Spherical Gaussian. A *spherical Gaussian* (SG) with *axis* $\xi \in S^2$ and *sharpness* $\lambda \in \mathbb{R}$ is given by $G(\omega; \xi, \lambda) = \exp(\lambda(\omega \cdot \xi - 1))$. The *mass function* $M(\lambda)$ is the integral of an SG with sharpness λ over the whole unit sphere S^2, and it is equal to $2\pi(1 - e^{-2\lambda})/\lambda$. Two properties of SGs make them attractive for shading calculations: (1) they can be easily rotated, and (2) the product of two SGs is another SG.

Tsai and Shih [2006] use a sum of SGs to represent the product between a factored part of the BRDF and the visibility function. However, we shall demonstrate in Section 6 that their scheme does not work with the highly specular BCSDF necessary for cloth models. We avoid this problem by using the sum of SGs to represent only the visibility function and approximating the shading integral analytically.

Wang et al. [2009] use SGs to represent glossy BSDFs and introduce the spherical signed distance function (SSDF) as a representation of visibility. Later works adapt SG-based representations to dynamic scenes [Iwasaki et al. 2012], improve shadowing [Zhou et al. 2013], and incorporate anisotropy [Xu et al. 2013]. We adopt Wang et al.'s SSDF, but we discovered that it does not work well when the SG light is soft, so we transition to the sum-of-SG visibility representation in that case.

3 OVERVIEW

Multiple scattering is responsible for much of the appearance of fabrics, but simulating it with path tracing is expensive. Since its effects are quite smooth, we aggressively approximate them by precomputing the indirect radiance distribution.

Although global interreflection between distant surfaces is important, this paper focuses on local multiple scattering *within* the fabric volume. We also assume both the camera and the light sources are above the fabric, ignoring cases involving the side and bottom faces. Our algorithm could be adapted to handle these cases.

Figure 3 gives an overview of our algorithm. We rely on two main types of precomputed functions to accelerate rendering:

- The *segment visibility function* (SVF) encodes whether a ray starting from a given fiber segment escapes the fabric volume in any given direction.
- The *incoming indirect radiance transfer function* (IIRTF) encapsulates how indirect illumination to a point in the fabric volume depends on the illumination to the fabric as a whole.

While there is one SVF associated with each fiber segment, we precompute the average of the IIRTF over *cells*—further subdivisions of exemplar blocks that may contain multiple fiber segments.

At rendering time, we must first intersect eye rays with the fabric. On the CPU, this involves casting rays through shell maps [Porumbescu et al. 2005]. While an equivalent process exists for the GPU [Jeschke et al. 2007], Section 5.1 discusses our simpler volume ray casting approach. After identifying the hit point, we compute the out-scattered light, splitting it into single and multiple scattering components.

For single scattering, we compute the triple product integral between the SVF, the radiance from the light source, and the BCSDF. This process is non-trivial for spherical Gaussian lights, and we show that our new way of representing the SVF yields good approximations over the whole range of SG sharpness. The integral,

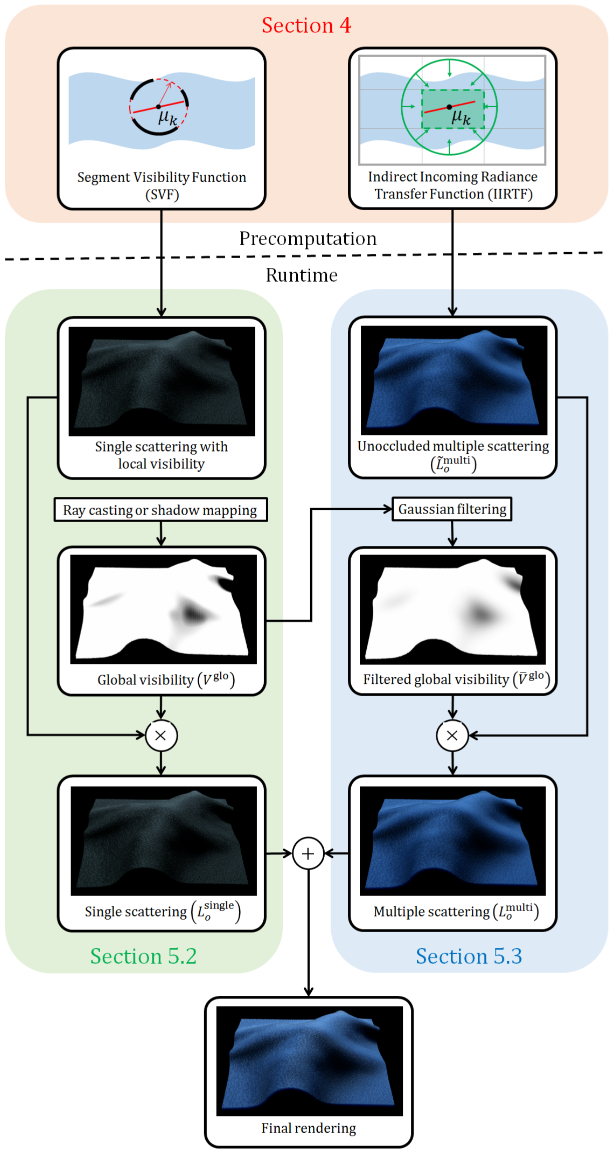

Section 4
μ_k
Segment Visibility Function (SVF)
μ_k
Indirect Incoming Radiance Transfer Function (IIRTF)
Precomputation
Runtime
Single scattering with local visibility
Unoccluded multiple scattering ($\tilde{L}_o^{\text{multi}}$)
Ray casting or shadow mapping
Gaussian filtering
Global visibility (V^{glo})
Filtered global visibility ($\bar{V}^{\text{glo}}$)
×
×
+
Single scattering (L_o^{single})
Multiple scattering (L_o^{multi})
Section 5.2
Section 5.3
Final rendering

Fig. 3. An overview of our algorithm.

however, only accounts for occlusion by nearby fiber microgeometry, so we incorporate macro-scale occlusion by other geometry using ray casting or shadow mapping against the fabric's shell. The single scattering process is discussed in Section 5.2.

For multiple scattering, we evaluate the IIRTF with the light source as an argument and convolve the result with the BCSDF to obtain the multiple scattering response to illumination unoccluded by macroscale geometry. To incorporate shadowing and to simulate subsurface scattering effects, we filter the shadow signal used in the single scattering computation with a spatial kernel, and scale the multiple scattering response by the result. Section 5.3 covers multiple scattering computation.

4 PRECOMPUTATION AND PARAMETERS

The first phase of our algorithm is a series of precomputations carried out once for each type of fabric (that is, once per exemplar and weave pattern) on a large, flat section of fabric synthesized in the same way as the material to be rendered.

4.1 Segment Visibility Function

Light occlusion by local fiber microgeometry is approximated using the precomputed segment visibility function $V_k : S^2 \to \mathbb{R}$ associated with each fiber segment k. $V_k(\omega) = 1$ if the infinite ray originating from μ_k in direction ω does not hit any fiber other than the one segment k is a part of, and $V_k(\omega) = 0$ otherwise.

We use two representations for V_k and choose between them based on whether the fabric is being shaded under a high or low frequency light source. Under high frequency lights (directional lights and SG lights with $\lambda \geq 200$), we use the *spherical signed distance function* (SSDF) [Wang et al. 2009], denoted by $d_k(\omega)$. Under low frequency lights (SG lights with $\lambda \leq 100$), we represent visibility with a weighted sum of (normalized) SGs:

$$V_k(\omega) \approx V_k^{\text{ssg}}(\omega) = \sum_{q=1}^{Q} w_{k,q} \frac{G(\omega; \xi_{k,q}, \lambda_{\text{svf}})}{M(\lambda_{\text{svf}})}.$$

where the axes $\xi_{k,q}$ are derived from the spherical Fibonacci point sets [Marques et al. 2013], and all the SGs in the representation have the same sharpness λ_{svf}. When the light's sharpness is between 100 and 200, we interpolate between the results yielded by the two representations. Figure 4 contains images of the SVF of a fiber segment and its representations.

SSDF implementation. To compute the SSDF of Segment k, we locate the copy of the exemplar block containing it that is the nearest to the center of the flat fabric, so as to avoid any boundary effects. With ray casting, we then render a 128×128 binary visibility image over the (θ, ϕ)-parameterization of the sphere and compute the SSDF from the resulting image by finding, for each pixel, the closest pixel having the opposite value. The whole collection of SSDFs is compressed with PCA using 48 principal components.

Sum of SGs implementation. The number of SGs, Q, is chosen to be 48, the same as the number of PCA components of the SSDFs. As discussed in Section 5.2.2, the sharpness parameter, λ_{svf}, should be at least 100, and we use $\lambda_{\text{svf}} = 128$ for all the renderings in this paper. The axes $\xi_{k,q}$ are derived them from the spherical Fibonacci point sets:

$$\xi_{k,q} = R_k \xi'_q, \qquad \xi'_q = (r_q \cos(\phi_q), r_q \sin(\phi_q), z_q),$$
$$r_q = \sqrt{1 - z_q^2}, \qquad \phi_q = 4\pi q/(1 + \sqrt{5}), \qquad z_q = 1 - (q-1)/Q$$

where R_k is a random rotation per segment around the z-axis to mask aliasing artifacts that might arise from using the same set of axes for all segments. To compute the weights $w_{k,q}$, we utilize the visibility image used to compute the SSDFs and set $w_{k,q}$ to the total solid angle of all the visible pixels closer to $\xi_{k,q}$ than any other axis.

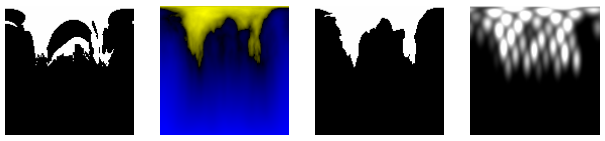
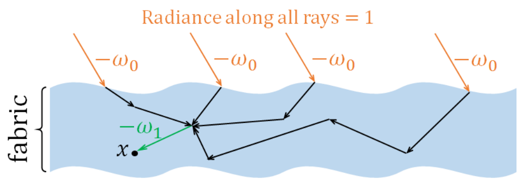


Fig. 4. The segment visibility function of a segment in the Silk fabric and its approximate representations. In Mercator projection, (a) visibility as a 128 × 128 binary image; (b) the PCA-compressed SSDF (yellow = positive, blue = negative); (c) the sign of the SSDF as a binary image; and (d) the sum of SGs representation.

4.2 Indirect Incoming Radiance Transfer Function

We use the IIRTF to convert the raw energy from the light sources to the incoming radiance at a point of interest *after* the light scatters one or more times from fibers. We compute the radiance due to local multiple scattering by convolving this incoming radiance with the BCSDF. Formally, the IIRTF $T(\omega_0, \omega_1, x)$ equals the incoming radiance arriving at point x from direction $-\omega_1$ through paths that (1) originate from a directional light source that emits unit radiance in direction $-\omega_0$ and (2) contain at least one point on a fabric fiber.



To make sure an IIRTF accurately describes the illumination in all instances of its cell in the fabric, we need a *regularity* assumption: for all instances of a given cell, the weave pattern in neighboring cells should be the same. This is obviously satisfied when the weave pattern across the whole fabric is the same as in the exemplar, but the method can also be applied to fabrics with varying weave patterns as long as local neighborhoods can be matched.

Working in the spherical harmonics domain, we represent the IIRTF—which transforms the SH expansion of a scalar function on S^2 to the expansion of another scalar function on S^2—as a matrix A_x of entries $a_x[\cdot,\cdot]$ where

$$a_x[j_0, j_1] = \int_{S^2}\int_{S^2} T(\omega_0, \omega_1, x) Y_{j_0}(\omega_0) Y_{j_1}(\omega_1)\, \mathrm{d}\omega_0 \mathrm{d}\omega_1.$$

Computation. Due to the IIRTF's size, we cannot afford to store one per each fiber. Instead, we divide exemplar blocks into *cells* of equal size and average the IIRTF over each cell. The per-cell IIRTF is the average of the IIRTF at all segment midpoints inside the cell:

$$T(\omega_0, \omega_1, C) = \frac{1}{|\mathscr{C}|} \sum_{k \in \mathscr{C}} T(\omega_0, \omega_1, \mu_k).$$

Here, C is a cell, and $\mathscr{C}$ is the set of segments with midpoints in C. The coefficient $a_C[j_0, j_1]$ of the cell's transfer function can be estimated by sampling k, ω_0, and ω_1 independently and computing:

$$\frac{1}{|\mathscr{C}|} \frac{T(\omega_0, \omega_1, \mu_k) Y_{j_0}(\omega_0) Y_{j_1}(\omega_1)}{p(k) p(\omega_0) p(\omega_1)}.$$

We sample all k's with equal probability ($p(k) = 1/|\mathscr{C}|$), ω_0 uniformly from the upper hemisphere ($p(\omega_0) = 1/(2\pi)$), and ω_1 uniformly from the whole sphere ($p(\omega_1) = 1/(4\pi)$). The value $T(\omega_0, \omega_1, \mu_k)$ is estimated by path tracing. We use 50,000 paths per cell.

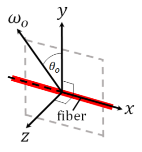
Choosing cell dimensions. We found that setting the side lengths of the cells as close as possible to the mean free path ℓ of a path traced through the fabric yielded consistently good results, with only a minor drop in visual quality with cells of size up to 2ℓ. The supplementary material discusses how we estimate ℓ.

Extension to bottom and side fabric surfaces. To take into account light scattered off the bottom fabric surface, we can simply sample light directions from below the fabric as well when precomputing the IIRTFs. To take into account the side faces, we must arrange the flat fabric used for precomputation so that each block is on an edge or a corner. This increases the space requirement by 8 times (4 possible corners and 4 edges). We ignore these cases in this paper as discussed previously, however.

4.3 Function Expansions into the SH Basis

We must compute the convolution between the BCSDF, the light source's radiance distribution, and the IIRTF, the last of which is expressed in the SH basis. This computation can be accelerated by expanding the first two into the same basis.

BCSDF. This expansion is defined in the coordinate system where the shading integral is performed: the fiber-based coordinate system used by Marschner et al. [2003]. The x-axis must coincide with the fiber segment's direction, but we are free to choose the y-axis so that the outgoing direction ω_o is in the xy-plane. Let us call this coordinate system the ω_o-space.



We precompute a table $C_S[\theta_o, j]$ where:

$$C_S[\theta_o, j] = \int_{S^2} Y_j(\omega_i) S(\omega_i, \omega_o) \cos\theta_i \, \mathrm{d}\omega_i$$

where S is the BCSDF proposed by Khungurn et al. [2015]. We use 512 equally-spaced values of θ_o from $[-\pi/2, \pi/2]$. For each value of θ_o, we compute the expansion up to the SH order of the IIRTF.

Spherical Gaussians. One of our goals is to render fabrics with local multiple scattering under SG lights, so we also express SGs in the SH basis. In particular, we precompute a table

$$C_G[\lambda, j] = \int_{S^2} \frac{G(\omega; (0,0,1), \lambda)}{M(\lambda)} Y_j(\omega)\, \mathrm{d}\omega,$$

which stores the SH coefficients of normalized SGs with various sharpness aligned with the z-axis. We store only the coefficients of the zonal harmonics because all others are zero. The supplementary material discusses how we choose the λ values and how we interpolate the entries.

4.4 Parameters

The parameters of our algorithm include the number of PCA components used to compress the SSDFs, the number and sharpness of SGs in the sum-of-SG visibility representation, the size of the fabric grid cells, and the SH order of the IIRTF.

There is another parameter, σ_{glo}, which is the standard deviation of the 2D Gaussian kernel used to approximate the effect of occlusion on local multiple scattering. Its role is discussed in Section 5.3.

5 RENDERING ALGORITHM

We now describe how to use the precomputed data to render fabrics. Suppose that the eye ray is along the direction $-\omega_o$. We first intersect the ray with the fabric (Section 5.1), giving the hit point x. We then compute the radiance leaving x in direction ω_o, $L_o(x, \omega_o) = L_o^{\text{single}}(x, \omega_o) + L_o^{\text{multi}}(x, \omega_o)$, in separate processes for single (Section 5.2) and multiple (Section 5.3) scattering.

5.1 Eye Ray–Fabric Intersection

On the CPU, primary visibility can be efficiently computed by tracing rays through the shell map and intersecting fibers stored in a spatial hierarchy. On the GPU, the more regular memory access of volume ray casting makes it a better approach, so we rasterize the fabric exemplar into a 3D volume. Each non-empty voxel stores the ID of the fiber segment whose midpoint is nearest to its center. The segment's direction and the ID of its BCSDF are stored in a separate texture indexed by segment ID. This volume is partitioned into blocks, tiled, and shell mapped as described in Section 2.

To render a shell-mapped fabric volume, we rasterize the front facing triangles of all tetrahedra in the shell. For each resulting fragment, we transform the eye ray into the flat fabric's space and sample the volume at a fixed number of linearly spaced points along the relevant ray segment to find the first non-empty voxel. (The number of points used can be found in Table 1.) If no such voxel is found, we discard the fragment. Otherwise, we save information about the hit point, including its shell texture coordinate and the fiber segment's ID, to a G-buffer for deferred shading.

5.2 Single Scattering

Having identified the hit point x on Fiber k, we are now ready to compute the out-scattered light from the fabric. This section describes its single scattering component. We first discuss the directional light case and then continue with the SG light case.

5.2.1 Directional Light. Let ω_d denote the direction toward the light source, and let us say that the radiance along ω_d is 1. The outgoing radiance due to single scattering is given by

$$L_o^{\text{single}}(x, \omega_o) = V(x, \omega_d) S(\omega_d, \omega_o) \cos\theta_d$$

where S is the BCSDF, and V is the visibility function. Evaluating V by tracing shadow rays is expensive because each ray has to be traced through many fibers. To avoid this, we split V into the *local visibility term*, which deals with occlusion by nearby fiber microgeometry, and the *global visibility term*, which deals with occlusion by macroscopic objects in the scene: $V(x, \omega_d) = V^{\text{loc}}(x, \omega_d) V^{\text{glo}}(x, \omega_d)$.

The local visibility term is approximated with the precomputed SVF: $V^{\text{loc}}(x, \omega_d) \approx V_k(\omega_d)$. In this case, we use the sign of the SSDF (Figure 4(c)) to determine visibility.

The global visibility term $V^{\text{glo}}(x, \omega_d)$ is approximated by tracing a shadow ray in direction ω_d, intersecting only against the fabric shell and other macroscopic objects in the scene.

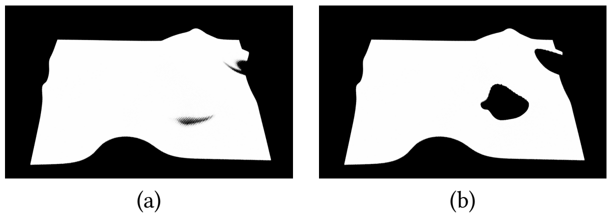
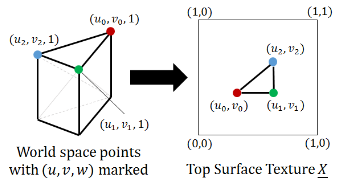


Fig. 5. Effects of the choice of shadow ray origin. (a) uses the hit point x (and skips the first intersection with the shell), and (b) uses the shell's top surface point that is directly above x.

The shadow ray starts at $\underline{x}$, the point on the top surface of the shell that is directly above the fiber hit point x. This choice enables computing global visibility on the GPU using a shadow map. It also prevents unintuitive hard shadows that are the result of tracing rays from x itself (See Figure 5). Our GPU implementation uses percentage-closer filtering (PCF) for shadow map anti-aliasing [Reeves et al. 1987], but many other more sophisticated techniques are available for this purpose.

On the CPU, $\underline{x}$ can be computed by a shell map lookup, which involves traversing the bounding volume hierarchy of the shell tetrahedra to identify the tetrahedron that contains the point with the given shell texture coordinate. On the GPU, we render a texture $\underline{X}[u, v]$ that maps the 2D shell texture coordinate (i.e. ignoring the depth component) to the world position of the top surface of the shell map. Given a hit point x, we can recover its shell texture coordinate (u, v, w) from the G-buffer. We then look up $\underline{X}[u, v]$ to determine $\underline{x}$.



Generalization to other types of light sources. On the CPU, the single scattering computation can be easily extended to any type of environment light source that can be efficiently sampled. We first sample the incoming direction ω_d and the radiance $\mathcal{L}(\omega_d)$ along it. We then compute the single scattering response and scale it by $\mathcal{L}(\omega_d)/p(\omega_d)$ where $p(\omega_d)$ is the probability of sampling ω_d. The definition of visibility and incoming light may be changed to accommodate point lights and, by extension, area light sources.

5.2.2 Spherical Gaussian Light. While random sampling can convert any arbitrary environment light—including any SG light—into a directional light, it does not work well on the GPU because of the lack of native support for tracing arbitrary shadow rays. Moreover, the approach will yield noisy renderings, particularly when the support of the SG light is large. Our goal is to design a GPU-friendly algorithm that shades micro-appearance models under SG lights *without noise.* We achieve this by exploiting the structure of both the precomputed visibility and the BCSDF.

Let the scene be illuminated by a single normalized SG light $G(\omega; \xi, \lambda)/M(\lambda)$. The single scattering component is given by:

$$\int_{S^2} V^{\text{glo}}(\omega_i) V^{\text{loc}}(\omega_i) \frac{G(\omega_i; \xi, \lambda)}{M(\lambda)} S(\omega_i, \omega_o) \cos\theta_i \, \mathrm{d}\omega_i. \tag{1}$$

We will first describe how we approximate global visibility under SG lights to simplify the problem. We will then discuss how to compute single scattering while taking into account the complex occlusion by nearby fibers.

Global visibility under SG lights. We recognize that accurate shadowing under area lights, including SG lights, is a challenging problem that currently has no solution without significant compromises. Approaches based on SSDFs [Wang et al. 2009; Zhou et al. 2013] require expensive precomputation of the SSDFs of macroscale geometry, making them impractical in rendering cloth animation. The integral SG approach [Iwasaki et al. 2012] requires rasterizing thin shell meshes at each fragment and so does not scale well in our setting. As a result, we settle on plausible shadow computation through percentage-closer soft shadow (PCSS) mapping [Fernando 2005], which scalably handles complex, changing geometry while producing visually pleasing shadows. Recent techniques [Annen et al. 2008; Shen et al. 2013; Yang et al. 2010] speed up PCSS by enabling prefiltering. However, we only use the original PCSS in our implementation. One drawback of PCSS is the need to determine parameters such as the sizes and positions of the light sources. We picked these parameters manually.

Specifically, we compute the global visibility term $V^{\text{glo}}(x)$ by PCSS. The global visibility term is then used to scale the power of the SG light down without changing the distribution. Namely, Equation (1) becomes:

$$\frac{V^{\text{glo}}(x)}{M(\lambda)} \int_{S^2} V^{\text{loc}}(x, \omega_i) G(\omega_i; \xi, \lambda) S(\omega_i, \omega_o) \cos\theta_i \, \mathrm{d}\omega_i.$$

Our problem thus reduces to the triple product integral between the local visibility, an SG, and the BCSDF.

We first describe the solution to the above problem with an approximation to the integral when the SG light is unoccluded and sharp ($\lambda \geq 100$). We then discuss how to use our two SVF representations to incorporate local visibility into the integral.

Unoccluded, sharp SG lights. The BCSDF we use is a sum of two terms: $S(\omega_i, \omega_o) = S_R(\omega_i, \omega_o) + S_{TT}(\omega_i, \omega_o)$, which represent light that reflects off the fiber surface and that transmits through the fiber, respectively [Khungurn et al. 2015]. In the summary, we show that, when $\lambda \geq 100$, we may approximate its convolution with the (unnormalized) SG with:

$$\begin{aligned} &\int_{S^2} G(\omega_i; \xi, \lambda) S(\omega_i, \omega_o) \cos\theta_i \, \mathrm{d}\omega_i \\ &\approx a_R \frac{B_R(\theta_R, \lambda\cos\theta')}{2} \sqrt{\frac{\pi}{\lambda_R}} \Big[\operatorname{erf}(\sqrt{\lambda_R}(\theta_i - \theta_R))\Big]_{-\pi/2}^{\pi/2} \qquad (2) \\ &+ a_{TT} \frac{B_{TT}(\theta_{TT}, \lambda\cos\theta', \phi')}{2} \sqrt{\frac{\pi}{\lambda_{TT}}} \Big[\operatorname{erf}(\sqrt{\lambda_{TT}}(\theta_i - \theta_{TT}))\Big]_{-\pi/2}^{\pi/2} \end{aligned}$$

where (θ', ϕ') is the fiber-based spherical coordinate of ξ, $\lambda_R = \beta_R^{-2}/2 + \lambda/2$,

$$\theta_R = \frac{-\beta_R^2 \theta_o + \lambda\theta'}{\beta_R^{-2} + \lambda}, \qquad a_R = \exp\left(-\frac{\beta_R^{-2}\lambda}{\beta_R^{-2} + \lambda} \frac{(\theta_o + \theta')^2}{2}\right),$$

and β_R is the standard deviation of the Gaussians in the longitudinal scattering functions of S_R. Variables involving the TT terms are defined similarly. The appendix contains the exact forms of the B_R and B_{TT} functions, and the supplementary material contains derivations. Let us denote the RHS of (2) with $\Gamma(\xi, \lambda, \omega_o)$.

Local visibility under soft SG lights. The approximation Γ only works with unoccluded and sharp SG lights. To shade a soft SG light occluded by nearby fiber segments, we use the fact that the product of two SGs reduces to an SG, which implies that the sum-of-SGs representation of the SVF can "break" the soft SG light into many sharp ones:

$$\begin{aligned} V^{\text{loc}}(x, \omega_i) G(\omega_i; \xi, \lambda) &\approx V_k^{\text{ssg}}(x, \omega_i) G(\omega_i; \xi, \lambda) \\ &= \sum_{q=1}^{Q} \frac{w_q}{M(\lambda_{\text{svf}})} G(\omega_i; \xi_{k,q}, \lambda_{\text{svf}}) G(\omega_i; \xi, \lambda) \\ &= \sum_{q=1}^{Q} \frac{w_q}{M(\lambda_{\text{svf}})} \frac{G(\omega_i; \widehat{\xi_{k,q}^{\text{sum}}}, \|\xi_{k,q}^{\text{sum}}\|)}{\exp(\lambda_{\text{svf}} + \lambda - \|\xi_{k,q}^{\text{sum}}\|)} \end{aligned}$$

where $\xi_{k,q}^{\text{sum}} = \lambda_{\text{svf}} \xi_{k,q} + \lambda\xi$, and $\widehat{\xi_{k,q}^{\text{sum}}} = \xi_{k,q}^{\text{sum}} / \|\xi_{k,q}^{\text{sum}}\|$. So,

$$\begin{aligned} &\int_{S^2} V_k^{\text{loc}}(x, \omega_i) G(\omega_i; \xi, \lambda) S(\omega_i, \omega_o) \cos\theta_i \, \mathrm{d}\omega_i \\ &\approx \sum_{q=1}^{Q} \frac{w_q}{M(\lambda_{\text{svf}})} \frac{\Gamma(\widehat{\xi_{k,q}^{\text{sum}}}, \|\xi_{k,q}^{\text{sum}}\|, \omega_o)}{\exp(\lambda_{\text{svf}} + \lambda - \|\xi_{k,q}^{\text{sum}}\|)}. \end{aligned}$$

A remarkable feature of the above expression is that, when Γ is inaccurate—that is, when $\|\xi_{k,q}^{\text{sum}}\| < 100$—it is divided by $\exp(\lambda_{\text{svf}} + \lambda - \|\xi_{k,q}^{\text{sum}}\|)$, which is large given than λ_{svf} is suitably large. With our choice of $\lambda_{\text{svf}} = 128$, we have that

$$\lambda_{\text{svf}} + \lambda - \|\xi_{k,q}^{\text{sum}}\| > 128 + \lambda - 100 \geq 28.$$

So, the erroneous approximation is scaled down by a factor of at least $e^{-28} \approx 7 \times 10^{-13}$, meaning that we aggressively suppress cases where Γ does not work well. In general, the sharpness λ_{svf} should become larger if more SGs are used in the sum so as to avoid over-blurring the visibility function.

A problem with the sum-of-SGs visibility representation is that it can produce renderings that are too bright when the SG light is sharp, especially in shadowed areas. The reason is that the representation is inherently soft: the Gaussians in the representation are continuous functions. As such, they yield positive function values at directions that are supposed to be occluded in the ground truth visibility function. Figure 6 illustrates this problem.

Local visibility under sharp SG lights. To combat the above problem, we gradually fall back to using the SSDFs to represent local visibility as λ increases from 100. Since the SG is sharp, we factor out the visibility term into a new one and approximate the double product integral with Γ as:

$$\begin{aligned} &\int_{S^2} V^{\text{loc}}(x, \omega_i) G(\omega_i; \xi, \lambda) S(\omega_i, \omega_o) \cos\theta_i \, \mathrm{d}\omega_i \\ &\approx \widetilde{V}^{\text{loc}}(x, \xi) \int_{S^2} G(\omega_i; \xi, \lambda) S(\omega_i, \omega_o) \cos\theta_i \, \mathrm{d}\omega_i \\ &\approx \widetilde{V}^{\text{loc}}(x, \xi) \Gamma(\xi, \lambda, \omega_o). \end{aligned}$$

The new visibility term $\widetilde{V}^{\text{loc}}(x, \xi)$ is similar to the SSDF calculation for the directional case, but with an error function serving as a

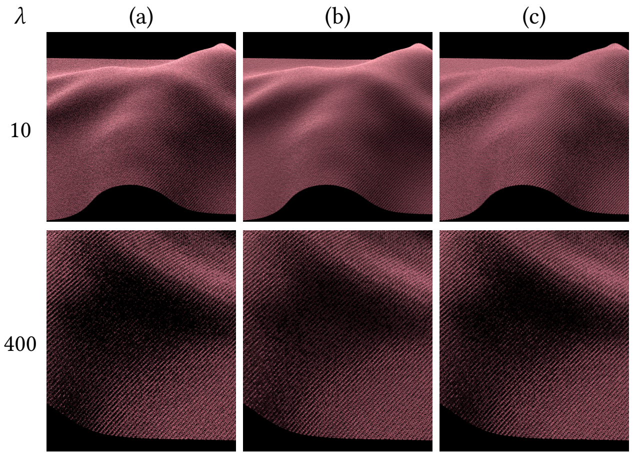


Fig. 6. Problematic regions of various single scattering approximations. All images only contain single scattering and do not take into account global visibility. Column (a) contains reference renderings produced by Monte Carlo integration. Column (b) uses the sum-of-SGs visibility representation. Column (c) uses the error-function-based visibility term coupled with the sharp SG light approximation, both discussed in Section 5.2.2.

smoothed step function:

$$\widetilde{V}^{\text{loc}}(x,\xi) = \frac{1}{2}\left[\text{erf}\left(\sqrt{\frac{\lambda}{2}}d_k(\xi)\right) + 1\right]$$

where d_k is the SSDF of the fiber segment. The standard deviation of the error function is $\sqrt{2/\lambda}$, motivated by the convolution of a sharp SG with a straight edge visibility function. We note that the aforementioned term is different from the one used by Wang et al. [2009], which is also a sigmoid-like function of $d_k(\xi)$ but has a more complicated form. Our term is simpler and already achieves good results when $\lambda > 100$.

To make the transition between the sum-of-SGs and the SSDF smooth, we linearly interpolate the single scattering results of the two schemes when $\lambda \in [100, 200]$, using λ itself as the interpolation parameter.

According to the measurements available in the supplementary material, using the SSDF with the error-function-based visibility term can make the algorithm 1% to 3% slower than the directional light case on the GPU. However, using the sum-of-SGs visibility can make it up to 13% slower. As such, when interpolating between the two schemes, the extra cost is dominated by the sum-of-SGs. The worst slowdown we observed is 16%.

5.3 Multiple Scattering

While single scattering accounts for non-smooth variation in fabric appearance, most of the fabric color comes from multiple scattering. We approximate multiple scattering as a product between (1) the multiple scattering response $\widetilde{L}_o^{\text{multi}}(x,\omega_o)$ of the fabric to unoccluded illumination from the light source and (2) a visibility factor $\bar{V}^{\text{glo}}(x)$; i.e. $L_o^{\text{multi}}(x,\omega_o) = \widetilde{L}_o^{\text{multi}}(x,\omega_o)\bar{V}^{\text{glo}}(x)$. The former relies on the precomputed IIRTF, and the latter employs Gaussian filtering of the global visibility discussed in the last section. We will now discuss the terms in order.

5.3.1 *Multiple Scattering Response to Unoccluded Light.*

Directional light. Recall that we assume a directional light that emits unit radiance in direction $-\omega_d$. If the light is not occluded in the neighborhood of x, then it induces the incoming radiance field $T(\omega_d,\omega_i,x)$ around x, and the outgoing light due to multiple scattering is:

$$\widetilde{L}_o^{\text{multi}}(x,\omega_o) = \int_{S^2} T(\omega_d,\omega_i,x)S(\omega_i,\omega_o)\cos\theta_i \,\mathrm{d}\omega_i.$$

After identifying the cell C containing the midpoint of the fiber segment on which x lies, we approximate $T(\cdot,\cdot,x)$ using the precomputed IIRTF matrix A_C. The convolution above can be computed as a dot product of SH coefficients. Evaluating the IIRTF involves projecting the directional light into SH basis by evaluating the vector $c_d = (Y_0(\omega_d), Y_1(\omega_d), \dots)$ and then multiplying it with the IIRTF matrix to obtain the vector $c_T = A_C c_d$. Using the precomputed table of BCSDF expansion, we have $S(\omega_i,\omega_o)\cos\theta_i \approx \sum_j C_S[\theta_o, j]Y(\omega_i)$. Let $c_S = (C_S[\theta_o,0], C_S[\theta_o,1], \dots)$. Conceptually, $\widetilde{L}_o^{\text{multi}}(x,\omega_o)$ is the dot product between c_S and c_T.

However, we cannot compute the dot product directly because the BCSDF's expansion is defined in the ω_o-space, but the IIRTF's expansion and thus c_T are defined in the fabric's object space. To solve this problem, we transform c_T into the ω_o-space. Let R_{ω_o} be the rotation matrix that transforms spherical harmonics expansion from the fabric's object space to the ω_o-space. Then, we have that $\widetilde{L}_o^{\text{multi}}(x,\omega_o) = c_S \cdot (R_{\omega_o} c_T)$. We compute the matrix using the technique described by Pinchon and Hoggan [2007].

Spherical Gaussian light. The computation is essentially unchanged. To compute the fabric's response to the SG light $G(\omega_i;\xi,\lambda)$, we look up the precomputed SG expansion table to get the vector $c_G = (C_G[\lambda,0], C_G[\lambda,1], \cdots)$, which represents the expansion of the SG with axis $(0,0,1)$ and sharpness λ. We then rotate the coefficient vector by a rotation R_ξ to align the axis with ξ. The rest of the process then applies. We multiply the rotated coefficient with the IIRTF matrix, rotate the result to the ω_o-space, and dot the rotated result with the SH expansion of the BCSDF: $\widetilde{L}_o^{\text{multi}}(x,\omega_o) = c_S \cdot (R_{\omega_o} A_C R_\xi c_G)$.

5.3.2 *Visibility for Multiple Scattering.*

We now estimate the effect of occlusion on multiple scattering. Since the IIRTF, by definition, has taken into account occlusion by local geometry, we only need to deal with occlusion by macroscopic geometry. We observe that, around shadowed areas on a piece of cloth, the shadow is not sharp due to light that propagates through the fabric volume into the occluded area; i.e., cloth exhibits subsurface-scattering-like behavior. We approximate this effect by multiplying the response-to-unoccluded-light term with a kernel-smoothed global visibility over the area near the shaded point x:

$$\bar{V}^{\text{glo}}(x,\omega_d) = \frac{\int_A K(x,x')V^{\text{glo}}(x',\omega_d)\,\mathrm{d}x'}{\int_A K(x,x')\,\mathrm{d}x'}$$

where A is the area of the (flat, before shell mapping) fabric surface, and K is the kernel function that depends only on the distance between x and x' in the fabric's plane. The global visibility term V^{glo} is computed by ray tracing or standard shadow mapping in the

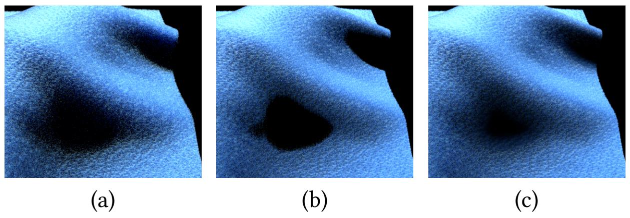


Fig. 7. The effect of the multiple scattering visibility term $\bar{V}^{\text{glo}}$. We show renderings by (a) path tracing, (b) using the (hard) global visibility term V^{glo} to scale down the multiple scattering response, and (c) using the (soft) global visibility term $\bar{V}^{\text{glo}}$ for the same purpose. Exposure of 4 is used to highlight the difference between shadowed and unshadowed areas. Notice that the shadowed regions in (b) have sharp edges, while those in (a) and (c) have softer edges.

directional light case and by PCSS in the SG light case. The effects of the visibility term can be seen in Figure 7.

We choose K to be a 2D Gaussian kernel (with standard deviation σ_{glo}) because it enables efficient implementations. How we implement filtering depends on the target hardware.

On the CPU, we sample x' according to the 2D Gaussian distribution in the flat fabric's space and use the global visibility term $V^{\text{glo}}(x', \omega_d)$ as the unbiased estimate of $\bar{V}^{\text{glo}}(x, \omega_o)$.

On the GPU, we compute the visibility texture $V[u,v] := V^{\text{glo}}(\underline{\mathrm{X}}[u,v])$ by looking up the shadow map at each position stored in the top surface position texture previously discussed in Section 5.2.1. We can then filter V by a 2D Gaussian kernel corresponding to K to obtain the average visibility texture $\bar{V}$, which can be done efficiently because the Gaussian kernel is separable. The global visibility term $\bar{V}^{\text{glo}}(x)$ is simply a lookup into $\bar{V}$ using the shell texture coordinate of x. (However, our implementation actually performs Monte Carlo integration with 40 samples per fragment with the help of a random number texture.)

6 RESULTS

We implemented two versions of our algorithm according to the hardware they run on. The CPU version employs Monte Carlo sampling of the light source discussed at the end of Section 5.2.1 and does not use approximation specific to SG lights in Section 5.2.2. The GPU version employs all GPU-specific computation and the entirety of Section 5.2.2. The algorithms were implemented in Java and the OpenGL Shading Language (GLSL). We performed experiments on 8 fabrics, which are derived from micro CT scans except for the shot silks, which are procedural fiber models. Their details are given in Table 1.

We will first describe the effects of the two parameters that are the most important to the appearance of the rendered fabrics: the dimensions of the IIRTF cells and the SH order used in the IIRTF. After fixing these parameters, we compare our algorithm against other algorithms. We then show that our algorithm can be used to effectively render a simple form of patterned cloth. Lastly, we discuss its limitations.

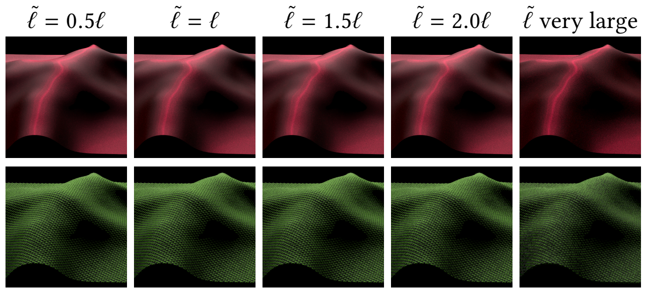


Fig. 8. The effects of the dimensions of IIRTF cells on renderings of the Silk (top) and the 2/3 Satin fabrics (bottom).

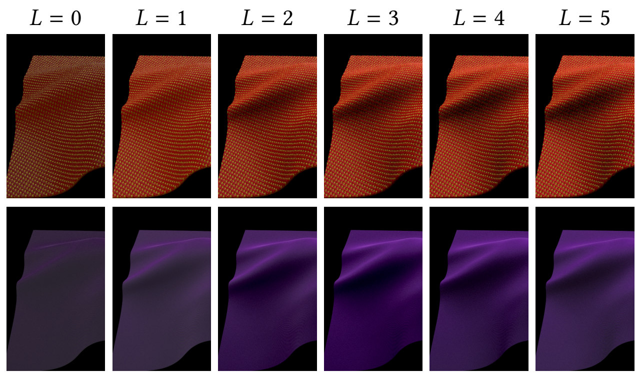


Fig. 9. Effects of the IIRTF's SH order, denoted by L, on renderings of the 4/1 Satin (top) and Shot Silk A (bottom) fabrics.

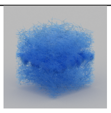
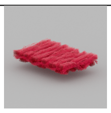
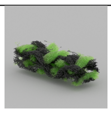
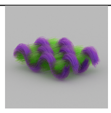
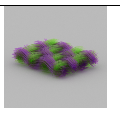
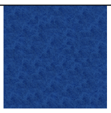
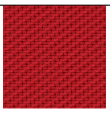
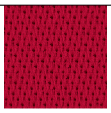
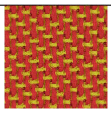
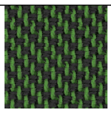
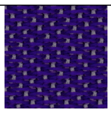
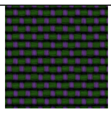
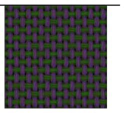
6.1 Effects of IIRTF Parameters

Cell dimensions. As discussed in Section 4.2, we subdivide an exemplar block into cells whose side lengths are as close as possible to a number which we now call the *target cell length* $\tilde{\ell}$. We consider a sequence of 5 target cell length values for each fabric. The first four are 0.5ℓ, ℓ, 1.5ℓ, and 2ℓ, where ℓ is the mean free path. We picked the fifth so that the numbers of cells in all dimensions are close to 1 in order to see effects of very coarse subdivisions. We then generated 5 IIRTF data according to the sequence and used them to render a draped piece of fabric with the CPU version of our algorithm. The SH order used was 5 in all renderings, which is the maximum SH order that we use in this paper. The renderings of the Silk and the 2/3 Satin fabrics are shown in Figure 8. The complete set of results are given in the supplementary material.

We observed that, in all fabrics, there are virtually no differences between the renderings of the target cell lengths of 0.5ℓ and ℓ, showing that the mean free path provides a good starting point for finding the right target cell length. The renderings generally become darker as cell dimensions become larger. This is because the IIRTFs of fiber segments deeper below the fabric surface are averaged with those near the top. Larger cell sizes also yield blockier, less smooth renderings in fabrics with multiple yarn colors.

For each fabric, we choose the coarsest subdivision that yields smooth renderings that are similar in color to the renderings of the finest subdivisions. The choices we made are listed in Table 1.

Table 1. The fabric models. More details, including the BCSDF parameters, can be found in the supplementary material.

	Fleece	Gabardine	Silk	4/1 Satin	2/3 Satin	1/4 Satin	Shot Silk A	Shot Silk B
Swatch								
Tiled								
#Fibers	31,091	8,377	7,681	21,953	16,357	18,507	2,005	2,005
#Segments	580,660	120,121	112,294	318,882	244,420	290,149	75,550	56,518
Fiber radius	0.00087 cm	0.00161 cm	0.00047 cm	0.00150 cm	0.00150 cm	0.00150 cm	0.00010 cm	0.00010 cm
Mean free path ℓ	0.03550 cm	0.00668 cm	0.00250 cm	0.01257 cm	0.01438 cm	0.01693 cm	0.00135 cm	0.00119 cm
Target cell length $\tilde{\ell}$	ℓ	2.0ℓ	2.0ℓ	2.0ℓ	1.5ℓ	1.5ℓ	2.0ℓ	2.0ℓ
SH Order used	4	4	5	4	4	5	5	5
IIRTF file size	7.87 MB	11.38 MB	9.01 MB	11.44 MB	10.52 MB	21.05 MB	25.97 MB	29.96 MB
IIRTF time*	5.67 m	5.25 m	2.19 m	8.44 m	5.03 m	5.57 m	9.75 m	9.43 m
SSDF file size	218.71 MB	50.05 MB	47.19 MB	122.84 MB	95.57 MB	112.32 MB	33.73 MB	26.76 MB
Sum-of-SGs file size	54.27 MB	11.23 MB	10.50 MB	29.80 MB	22.84 MB	27.12 MB	7.06 MB	5.28 MB
Visibility time*	79.79 m	15.51 m	15.07 m	41.33 m	31.25 m	37.90 m	7.98 m	7.18 m
#Samples/tetrahedron	32	32	32	32	32	32	64	64

*Precomputations were performed on a cluster of 5 machines having 192 cores.

SH order. We used the CPU version of our algorithm to render all the fabrics, varying the SH order of the IIRTF from 0 to 5. We show renderings of the 4/1 Satin and Shot Silk A fabrics in Figure 9. The complete set of results can be found in the supplementary material.

In general, as we increase the SH order, we see more "directionality." That is, the highlights become more defined and sharper, and shadows also become darker. Color changes start to stabilize after SH of order 3.

The fabrics can be divided into two groups based on their responses to the SH order. In the Fleece, Gabardine, 4/1 Satin, and 2/3 Satin fabrics, we observe smaller changes between consecutive pairs of SH orders as we move through higher SH orders. In this group, differences between renderings of SH order 4 and 5 are minor and limited to highlight structures. In the 1/4 Satin, the Silk, and the Shot Silk fabrics, however, significant changes can be observed between SH order 4 and 5. For this reason, we choose to use SH order 4 for the first group, and SH order 5 for the second group.

6.2 Comparison with Other PRT Schemes

We compare our algorithm against Tsai and Shih's scheme [2006] and the original PRT work of Sloan et al. [2002], which is representative of standard PRT schemes in which single scattering is not separated from multiple scattering.

Tsai and Shih. The PRT scheme proposed by Tsai and Shih can only approximate single scattering. It discretizes the BCSDF into a matrix and approximates the matrix with a low rank approximation obtained by the singular value decomposition (SVD). The rank R of this approximation is an important parameter of the scheme, and the details of our implementation can be found in the supplementary material. The paper recommends using $R = 16$.

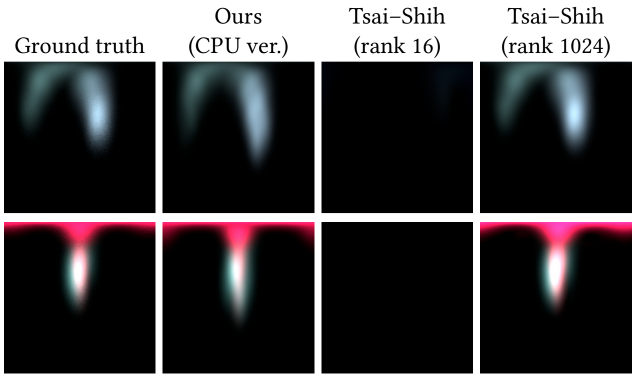


Fig. 10. Outgoing single scattered radiance from a Fleece fiber segment (top) and a Silk fiber segment (bottom) along two fixed directions (one for each type of fabric) when illuminated by an SG light ($\lambda = 50$) whose axis varies over the whole sphere of directions. The exposure was set to 8 to make the dim single scattering response bright enough to be seen clearly.

We rendered images that represent the response of a fiber segment in a flat piece of fabric to a single SG light. The viewing direction ω_o is fixed, but the axis of the SG light ξ is allowed to vary over the sphere of directions, which is flattened with Mercator projection. A pixel's value is equal to the triple product integral (1) where ξ is the direction that corresponds to the pixel. We show the images for a Fleece fiber segment and a Silk fiber segment in Figure 10. The ground truth images were computed with Monte Carlo integration

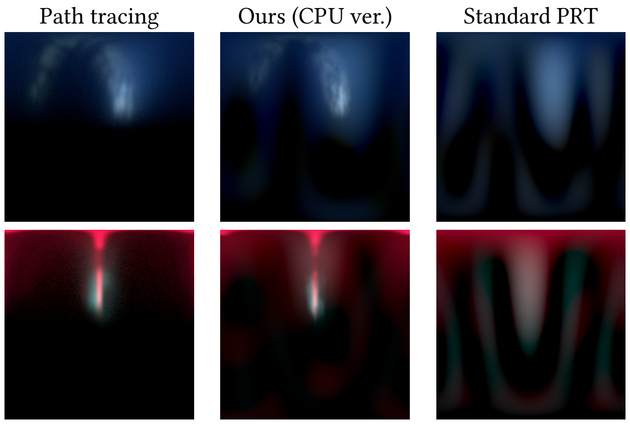


Fig. 11. Outgoing radiance from a Felt fiber segment (top) and a Silk fiber segment (bottom) along two fixed directions (one for each type of fabric) when illuminated by an SG light ($\lambda = 400$) whose axis varies over the whole sphere of directions.

using 1024 samples per pixel. We show images where the SG light's sharpness is 50, which is in the sharpness range where our algorithm uses the sum-of-SGs visibility.

Images rendered by our algorithm and the Tsai–Shih scheme with full rank are similar to the ground truth images, with the Tsai–Shih results being more accurate than ours because it uses more SGs than our algorithm does. However, when only the first 16 SVD terms are used, the Tsai–Shih scheme produced images that are practically black. The reason that low rank approximation does not work well in these matrices might be because the BCSDF is highly specular: most of its energy is concentrated in a narrow band around a diagonal, making it hard to approximate well with a sum of a few rank 1 matrices. (See the supplementary material for a visual demonstration.) As a result, we conclude that the Tsai–Shih scheme does not work well with the BCSDF used in this paper.

Standard PRT. We modified the algorithm for computing the IIRTF in Section 4.2 so that (1) it also includes direct illumination from the directional light source, and (2) it computes a transfer function per each fiber segment instead of per each cell. The transfer function computed in this way is equivalent to the transfer matrix in the work of Sloan et al. [2002], allowing us to compare against the standard PRT approach. In Figure 11, we show images of outgoing radiance from a single fiber segment similar to those in Figure 10. Moreover, in Figure 12, we show renderings of the Silk and Shot Silk A fabrics produced by path tracing, our algorithm, and the standard PRT approach. In both figures, the fabrics are illuminated by single high-frequency light sources. Due to the low SH order, the standard PRT approach is unable to reproduce specular reflection (i.e., sharp highlights in the Silk fabric) and also introduces visible ringing artifacts (i.e., wrong colors in some areas of the Shot Silk A fabric). Our algorithm, on the other hand, is able to faithfully reproduce sharp specular highlights and approximate the multiple scattering component with much fewer ringing artifacts.

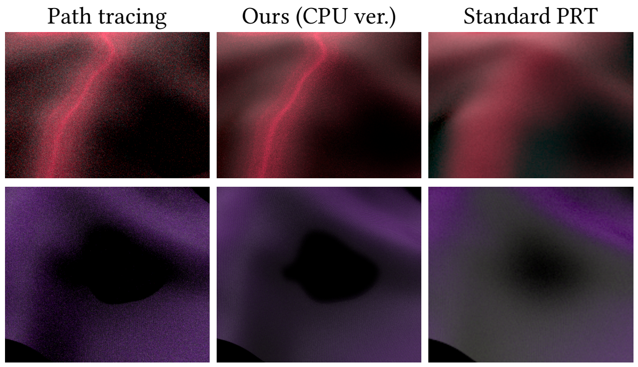


Fig. 12. Renderings of the Silk (top) and Shot Silk A (bottom) fabrics produced by path tracing, the CPU version of our algorithm, and the standard PRT approach.

6.3 Comparison with Other Fiber Assembly Rendering Algorithms

Quantitative match in flat configurations. We compare between dual scattering[1] and the two versions of our algorithm. We rendered flat pieces of the 8 fabrics under 492 scene configurations used to validate fitted models in [Khungurn et al. 2015]. We rendered each configuration as a 64×64 images with 128 samples per pixel. For the GPU version of our algorithm, we also vary the number of volume samples per shell tetrahedron (32, 64, and 128).

For each image rendered, we computed the average intensity, resulting in $3 \times 492 = 1{,}476$ values per algorithm and per fabric. The root mean squared errors (RMSE) of these values when compared to those produced by path tracing are graphed in Figure 13. The data show that, if the GPU version uses enough volume samples, both versions of our algorithm are more accurate than dual scattering. The result can be attributed to the assumptions that dual scattering (which was designed for rendering hair) makes—for example, that all fibers have the same BCSDF and nearby fibers are parallel to one another—which are violated in fabrics. The CPU version of our algorithm is generally more accurate than the GPU one because of its accurate ray intersection. As we increase the number of volume samples per tetrahedron, the GPU version becomes more accurate, except when rendering the Silk fabric, which might be because the rasterized Silk volume is the coarsest among all the volumes.

Renderings. We rendered the 8 fabrics in a draped configuration under 5 different lightings. To prevent algorithms from picking up illumination from underneath the fabric, we put a black mesh of the same shape as the shell underneath the fabrics. The images are rendered with 256 samples per pixel at resolution 1024×1024. We compare between four algorithms: path tracing, dual scattering, and both versions of our algorithm. Some of the renderings are available in Figure 16.

Our algorithm generally produced images whose colors are similar to path tracing references, while dual scattering yielded images that are generally darker. (For the Shot Silk fabrics, however, it

[1]See the details of our implementation of dual scattering in the supplementary material. We use the value 1.0 for all scattering density factors, but it still produced images that are not as bright as the path tracing references for most of the fabrics.

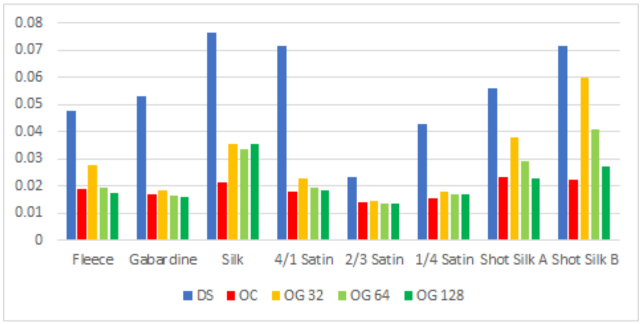


Fig. 13. RMSEs when compared to path tracing references of the average intensities of the 492 images rendered by dual scattering (DS), the CPU version of our algorithm (OC), and three runs of GPU version (OG) using 32, 64, and 128 volume samples per tetrahedron. The supplementary material contains numerical values of the RMSEs.

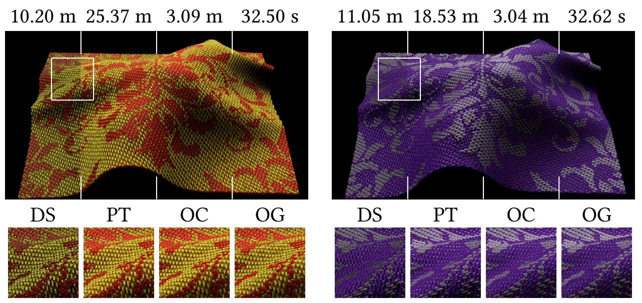


Fig. 14. Renderings of patterned damask fabrics constructed from the 4/1 Satin and 1/4 Satin exemplars created by dual scattering (DS), path tracing (PT), and the CPU and GPU version of our algorithm (OC and OG, respectively). The scenes are illuminated by an SG light with $\lambda = 150$. The supplementary material contains a more complete set of renderings and timing information.

overly amplified the green color. The renderings of our GPU algorithm under the SG light with $\lambda = 10$ are darker than those of other algorithms because PCSS overestimates shadows when the light source is large. Our algorithm also produced noticeable changes in highlights in all fabrics, especially in the Fleece and the Shot Silks. Subtleties in the shadows yarns cast on one another are missed in the Satin fabrics. and these deviations might be caused by low order spherical harmonics' inability to capture all the features of the incoming radiance field in the fabric volumes. We emphasize, though, that our renderings look plausible and have similar colors to the ground truth. Moreover, our GPU algorithm produced essentially noise-less images as it does not rely heavily on stochastic sampling.

We ran the first three algorithms on a cluster of 5 machines equipped with a total of 192 cores. Our GPU algorithm was run on an Nvidia GeForce GTX 980 graphics card. In Table 2, we report the wall clock time measured on the PC that control both the cluster and the GPU. The table clearly shows that our GPU algorithm could render in tens of seconds the images a compute cluster needs several minutes to render, all with a single commodity GPU. The supplementary material details our hardware settings and what the wall clock time includes.

Our CPU algorithm is the fastest among all the CPU-based algorithms, but its speedup over path tracing depends on the rendered fabrics. Path tracing is efficient in fabrics that result in low average path length. For example, the Silk has a strong reflective component, so light tends to reflects off it rather than going inside. Moreover, the 2/3 Satin has a black yarn, so Russian roulette tends to terminate paths early. Even in these cases, our CPU algorithm achieves a speedup of 2. For the other fabrics, light tends to remain in the fabric volume due to the fibers' strong transmission, so multiple scattering is more visually important for these fabrics. When rendering them, our algorithm achieves significant speedups. As we did not implement modular flux transfer (MFT), we could not compare to it directly. Instead, we compare against conservative estimates of MFT's running time, computed as described in Table 2. In the fabrics that are difficult for path tracing, our running times are 2 to 4 times better than MFT's estimates. We also include running times of dual scattering for completeness. However, as we did not implement the GPU version of dual scattering, they do not represent how efficient the algorithm can be.

6.4 Patterned Cloth

We now show that our approach can be used to render patterned cloth. Our method assumes that light transport in the cloth being rendered is the same as in the exemplar, which is strictly true only for cloth with a regular weave pattern. However, most patterned cloth is built up from a few regular weave patterns, and our algorithm can be used with minimal modifications for this type of pattern. For cloth that uses k different patterns, we require an exemplar for each of the patterns and an integer-valued image indicating which pattern is used in each block of the material. To render such a patterned cloth, we only need to determine, for each hit point, the exemplar associated with it, and use the appropriate precomputed data to shade the point.

We show renderings of patterned cloth in Figure 14. Because the precomputed data for each exemplar was created without knowledge of the other, they do not take into account the changes in multiple scattering and shadowing that occur at the borders between the two exemplars. However, our renderings already show colors that are comparable to the path tracing references.

6.5 Limitations

The memory usage and speed of our algorithm depend strongly on the size of the data used. In particular, the GPU implementation does not perform well when rendering dense fabrics: those whose mean free path is small relative to the exemplar's size. A short mean free path leads to large IIRTF data, which can thrash the GPU memory bandwidth or might not fit in GPU memory altogether. We note that this is a problem faced by any algorithm that performs volumetric precomputation on a uniform grid: dense material means appearance changes fast spatially, so such an algorithm needs a fine grid to be accurate.

Our algorithm yields blocky artifacts at zoom levels where fiber segments occupy multiple pixels because all fragments from the same segment use the same SVF and IIRTF. However, these artifacts are at the level of fiber segments, so they are not visible as long as the

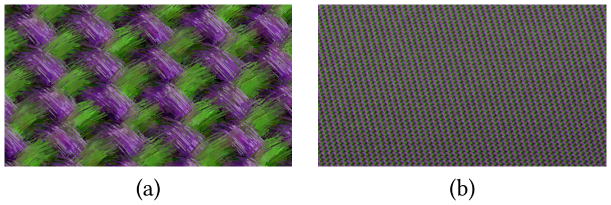


Fig. 15. The Shot Silk B fabric at two magnifications. (a) In a microscope view, which is beyond our intended range of application, it is plain to see that on each segment, all pixels in the same block have the same color. (b) However, at a magnification closer to what can be seen by the naked eye, the blockiness is invisible. The supplementary contains an animated version of (b), showing that the artifacts do not cause temporal flickering.

zoom level is not that of a microscope (see Figure 15). Moreover, they do not cause any flickering on animated fabrics (see supplementary videos).

7 CONCLUSION

We have described a GPU-friendly algorithm for approximate rendering of micro-appearance models of fabrics. It allows a commodity GPU to render, in tens of seconds, high-quality images with multiple scattering that a sizable CPU cluster needs to spend several minutes on. The efficiency gain is possible through decomposing the radiance computation into appropriate parts and using specialized precomputation for each. We employ the IIRTF to compute indirect illumination in a single step. We also present a scheme for approximating direct illumination from cloth fibers that can exploit both the structure of the fiber's scattering function and the spherical Gaussian light source, while accurately taking into account complex occlusion by nearby fibers. Our main contribution lies in identifying these representations.

Future directions. Our algorithm requires regular weave patterns or a limited form of patterned cloth created from them. A precomputation scheme that allows multiple scattering to be approximated in fabrics with arbitrary, complex weave patterns would expand its applicability. An efficient compression scheme for precomputed data is necessary to enable multiple fabrics to be rendered all at once. It is also interesting to see whether the sum-of-SGs visibility representation can be applied in other rendering contexts. Lastly, our algorithm's performance is limited by the need of high sample counts per pixel to avoid aliasing when the fabric is viewed from far away. A level-of-detail algorithms that account for internal multiple scattering is thus of great interest.

ACKNOWLEDGEMENTS

We thank Shuang Zhao for providing us with most of the fabric micro CT data, Eston Schweickart for the initial implementation of the yarn simulator used to generate the Shot Silk fabrics, and Fujun Luan for helping with the project. This research was funded by the National Science Foundation (grant IIS-1513967).

A EXACT FORMS OF B_R AND B_{TT} FUNCTIONS

In Section 5.2.2, the convolution between a sharp SG light and the BCSDF is written in terms of two functions, B_R and B_{TT}. The exact forms of the two functions contain the following parameters of the BCSDF proposed by Khungurn et al. [2015]:

- C_R: the color of the R mode,
- β_R: the longitudianal lobe width of the R mode,
- C_{TT}: the color of the TT mode,
- β_{TT}: the longitudinal lobe width of the TT mode, and
- γ_{TT}: the azimuthal lobe width of the TT mode.

With the parameters defined, B_R and B_{TT} are given below:

$$B_R(\theta_i,\lambda) = \frac{\mathscr{F}_R(\theta_i)\cos^2\theta_i}{\sqrt{2\pi}\beta_R\mathscr{G}(-\theta_i;\beta_R^{-2}/2)}\frac{I_0(\lambda\cos\theta_i)}{e^{\lambda\cos\theta_i}}$$

$$B_{TT}(\theta_i,\lambda,\phi) = \frac{(1-\mathscr{F}_R(\theta_i))C_{TT}\cos^2\theta_i}{\sqrt{2\pi}\beta_{TT}\mathscr{G}(-\theta_i;\beta_{TT}^{-2}/2)I_o(\gamma_{TT}^{-2})}\frac{I_0(\lambda_m(\theta_i,\lambda,\phi))}{e^{\lambda\cos\theta_i}}$$

where

$$\mathscr{F}_R(\theta_i) = C_R - (1-C_R)(1-\cos\theta_i)^5$$

$$\mathscr{G}(\mu;\lambda) = \sqrt{\frac{\lambda}{\pi}}\int_{-\pi/2}^{\pi/2} g(\theta;\mu,\lambda)\cos^2\theta\,\mathrm{d}\theta$$

$$\lambda_m(\theta_i,\lambda,\phi) = \sqrt{\gamma_{TT}^{-4} + \lambda^2\cos^2\theta_i - 2\gamma_{TT}^{-2}\lambda\cos\theta_i\cos\phi}$$

and I_0 is 0th modified Bessel function of the first kind. In our implementation, $\mathscr{G}(-\theta_i;\beta_R^{-2}/2)$ and $\mathscr{G}(-\theta_i;\beta_{TT}^{-2}/2)$ are tabulated while other terms are computed on the fly. The supplementary material contains the derivation of B_R and B_{TT}.

REFERENCES

Neeharika Adabala, Nadia Magnenat-Thalmann, and Guangzheng Fei. 2003. Visualization of Woven Cloth. In *Proceedings of the 14th Eurographics Workshop on Rendering (EGRW '03)*. Eurographics Association, 178–185.

Thomas Annen, Zhao Dong, Tom Mertens, Philippe Bekaert, Hans-Peter Seidel, and Jan Kautz. 2008. Real-time, all-frequency shadows in dynamic scenes. *ACM Trans. Graph.* 27, 3 (2008), 1–8.

Adrian Blumer, Jan Novák, Ralf Habel, Derek Nowrouzezahrai, and Wojciech Jarosz. 2016. Reduced Aggregate Scattering Operators for Path Tracing. *Computer Graphics Forum (Proceedings of Pacific Graphics)* 35, 7 (Oct. 2016). https://doi.org/10.1111/cgf.13043

Randima Fernando. 2005. Percentage-closer Soft Shadows. In *ACM SIGGRAPH 2005 Sketches (SIGGRAPH '05)*. ACM, New York, NY, USA, Article 35. https://doi.org/10.1145/1187112.1187153

Piti Irawan and Steve Marschner. 2012. Specular Reflection from Woven Cloth. *ACM Trans. Graph.* 31, 1, Article 11 (Feb. 2012), 20 pages.

Kei Iwasaki, Wataru Furuya, Yoshinori Dobashi, and Tomoyuki Nishita. 2012. Real-time Rendering of Dynamic Scenes Under All-frequency Lighting Using Integral Spherical Gaussian. *Comput. Graph. Forum* 31, 2pt4 (May 2012), 727–734. https://doi.org/10.1111/j.1467-8659.2012.03052.x

Kei Iwasaki, Kazutaka Mizutani, Yoshinori Dobashi, and Tomoyuki Nishita. 2014. Interactive Cloth Rendering of Microcylinder Appearance Model under Environment Lighting. *Computer Graphics Forum* (2014). https://doi.org/10.1111/cgf.12302

Stefan Jeschke, Stephan Mantler, and Michael Wimmer. 2007. Interactive Smooth and Curved Shell Mapping. In *Rendering Techniques*, Jan Kautz and Sumanta Pattanaik (Eds.). The Eurographics Association. https://doi.org/10.2312/EGWR/EGSR07/351-360

Pramook Khungurn, Daniel Schroeder, Shuang Zhao, Kavita Bala, and Steve Marschner. 2015. Matching Real Fabrics with Micro-Appearance Models. *ACM Trans. Graph.* 35, 1, Article 1 (Dec. 2015), 26 pages. https://doi.org/10.1145/2818648

Jaakko Lehtinen and Jan Kautz. 2003. Matrix Radiance Transfer. In *Proceedings of the 2003 Symposium on Interactive 3D Graphics (I3D '03)*. ACM, New York, NY, USA, 59–64. https://doi.org/10.1145/641480.641495

Bradford J. Loos, Lakulish Antani, Kenny Mitchell, Derek Nowrouzezahrai, Wojciech Jarosz, and Peter-Pike Sloan. 2011. Modular Radiance Transfer. In *Proceedings of the 2011 SIGGRAPH Asia Conference (SA '11)*. ACM, New York, NY, USA, Article 178, 10 pages. https://doi.org/10.1145/2024156.2024212

Ricardo Marques, Christian Bouville, Michael Ribardière, Luís Pualo Santos, and Kadi Bouatouch. 2013. Spherical Fibonacci Point Sets for Illumination Integrals. *Computer Graphics Forum* 32, 8 (2013), 134–143. https://doi.org/10.1111/cgf.12190

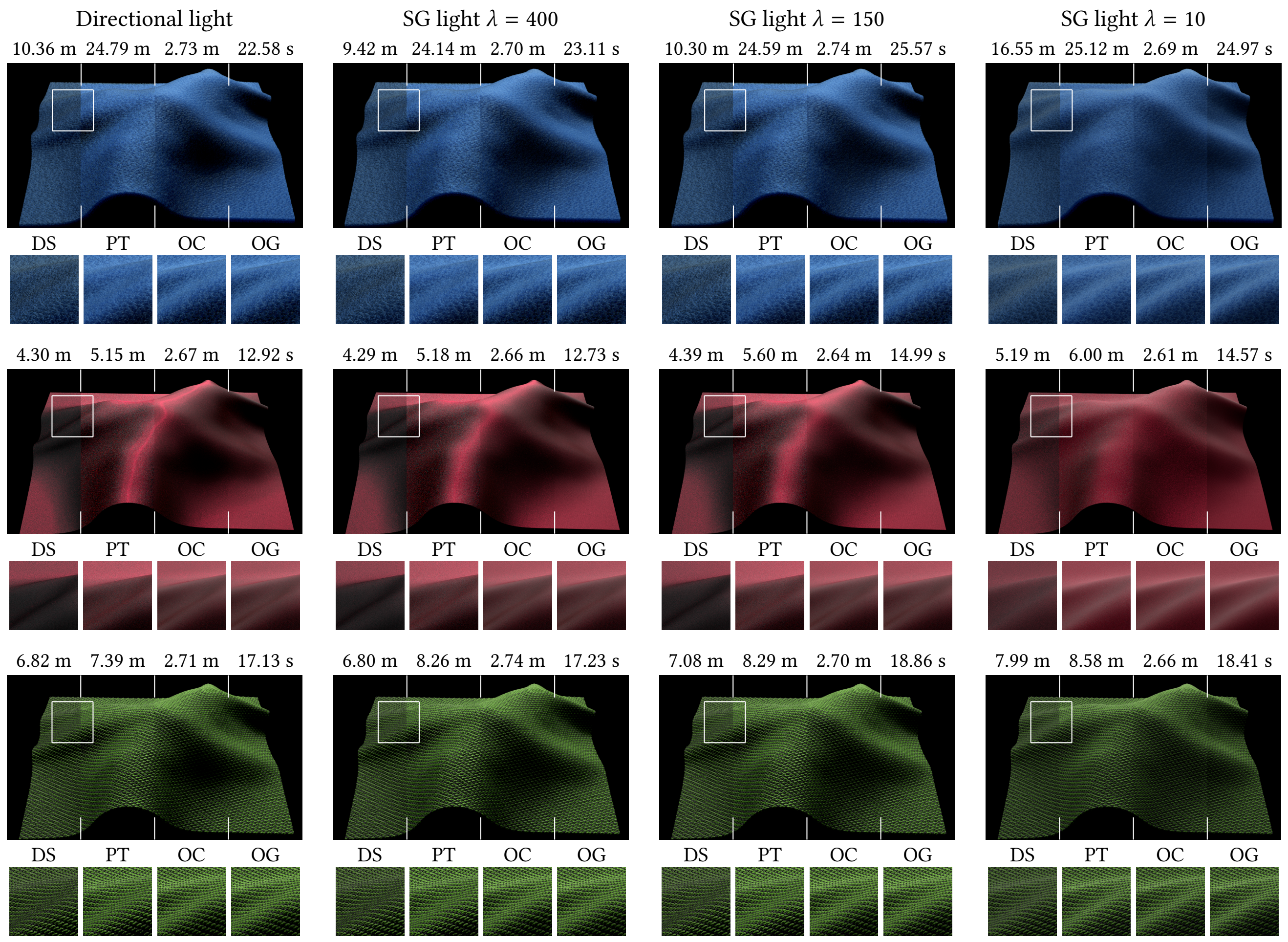

Directional light
10.36 m 24.79 m 2.73 m 22.58 s
SG light $\lambda = 400$
9.42 m 24.14 m 2.70 m 23.11 s
SG light $\lambda = 150$
10.30 m 24.59 m 2.74 m 25.57 s
SG light $\lambda = 10$
16.55 m 25.12 m 2.69 m 24.97 s
DS PT OC OG
4.30 m 5.15 m 2.67 m 12.92 s
4.29 m 5.18 m 2.66 m 12.73 s
4.39 m 5.60 m 2.64 m 14.99 s
5.19 m 6.00 m 2.61 m 14.57 s
6.82 m 7.39 m 2.71 m 17.13 s
6.80 m 8.26 m 2.74 m 17.23 s
7.08 m 8.29 m 2.70 m 18.86 s
7.99 m 8.58 m 2.66 m 18.41 s

Fig. 16. From top to bottom, renderings of the Fleece, Silk, and 2/3 Satin fabrics under 4 lighting configurations. We compare results by dual scattering (DS), path tracing (PT), and the CPU and GPU versions of our algorithms (OC and OG, respectively). The times on top of the renderings are the wall clock times used to render the full images. The supplementary material contains the complete set of results.

Stephen R. Marschner, Henrik Wann Jensen, Mike Cammarano, Steve Worley, and Pat Hanrahan. 2003. Light Scattering from Human Hair Fibers. *ACM Trans. Graph.* 22, 3 (July 2003), 780–791. https://doi.org/10.1145/882262.882345

Johannes Meng, Marios Papas, Ralf Habel, Carsten Dachsbacher, Steve Marschner, Markus Gross, and Wojciech Jarosz. 2015. Multi-scale Modeling and Rendering of Granular Materials. *ACM Trans. Graph.* 34, 4, Article 49 (July 2015), 13 pages. https://doi.org/10.1145/2766949

Jonathan T. Moon and Stephen R. Marschner. 2006. Simulating Multiple Scattering in Hair Using a Photon Mapping Approach. *ACM Trans. Graph.* 25, 3 (July 2006), 1067–1074. https://doi.org/10.1145/1141911.1141995

Jonathan T. Moon, Bruce Walter, and Steve Marschner. 2008. Efficient Multiple Scattering in Hair Using Spherical Harmonics. In *ACM SIGGRAPH 2008 Papers (SIGGRAPH '08)*. ACM, New York, NY, USA, Article 31, 7 pages. https://doi.org/10.1145/1399504.1360630

Jonathan T. Moon, Bruce Walter, and Stephen R. Marschner. 2007. Rendering Discrete Random Media Using Precomputed Scattering Solutions. In *Proceedings of the 18th Eurographics Conference on Rendering Techniques (EGSR'07)*. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 231–242. https://doi.org/10.2312/EGWR/EGSR07/231-242

Thomas Müller, Marios Papas, Markus Gross, Wojciech Jarosz, and Jan Novák. 2016. Efficient Rendering of Heterogeneous Polydisperse Granular Media. *ACM Trans. Graph.* 35, 6, Article 168 (Nov. 2016), 14 pages. https://doi.org/10.1145/2980179.2982429

Ren Ng, Ravi Ramamoorthi, and Pat Hanrahan. 2004. Triple Product Wavelet Integrals for All-frequency Relighting. *ACM Trans. Graph.* 23, 3 (Aug. 2004), 477–487. https://doi.org/10.1145/1015706.1015749

Minghao Pan, Rui Wang Xinguo Liu, Qunsheng Peng, and Hujun Bao. 2007. Precomputed radiance transfer field for rendering interreflections in dynamic scenes. In *Computer Graphics Forum*, Vol. 26. Wiley Online Library, 485–493.

Didier Pinchon and Philip E Hoggan. 2007. Rotation matrices for real spherical harmonics: general rotations of atomic orbitals in space-fixed axes. *Journal of Physics A: Mathematical and Theoretical* 40, 7 (2007), 1597. http://stacks.iop.org/1751-8121/40/i=7/a=011

Serban D. Porumbescu, Brian Budge, Louis Feng, and Kenneth I. Joy. 2005. Shell Maps. In *ACM SIGGRAPH 2005 Papers (SIGGRAPH '05)*. ACM, New York, NY, USA, 626–633. https://doi.org/10.1145/1186822.1073239

Ravi Ramamoorthi. 2009. *Precomputation-Based Rendering*. NOW Publishers Inc. http://graphics.cs.berkeley.edu/papers/Ramamoorthi-PBR-2009-04/

William T. Reeves, David H. Salesin, and Robert L. Cook. 1987. Rendering Antialiased Shadows with Depth Maps. *SIGGRAPH Comput. Graph.* 21, 4 (Aug. 1987), 283–291. https://doi.org/10.1145/37402.37435

Zhong Ren, Kun Zhou, Tengfei Li, Wei Hua, and Baining Guo. 2010. Interactive Hair Rendering Under Environment Lighting. In *ACM SIGGRAPH 2010 Papers (SIGGRAPH '10)*. ACM, New York, NY, USA, Article 55, 8 pages. https://doi.org/10.1145/1833349.1778792

Iman Sadeghi, Oleg Bisker, Joachim De Deken, and Henrik Wann Jensen. 2013. A practical microcylinder appearance model for cloth rendering. *ACM Trans. Graph.* 32, 2, Article 14 (April 2013), 12 pages. https://doi.org/10.1145/2451236.2451240

Mirko Sattler, Ralf Sarlette, and Reinhard Klein. 2003. Efficient and Realistic Visualization of Cloth. In *Eurographics Symposium on Rendering 2003*.

Table 2. The wall clock time used to render the draped fabrics under an SG light with $\lambda = 150$.

	Fleece		Gabardine		Silk		4/1 Satin	
	Time	Speedup	Time	Speedup	Time	Speedup	Time	Speedup
PT	24.59 m	1.00x	16.21 m	1.00x	5.60 m	1.00x	25.74 m	1.00x
DS	10.30 m	2.39x	6.88 m	2.36x	4.39 m	1.28x	9.94 m	2.59x
MFT*	11.88 m	2.07x	7.70 m	2.10x	5.09 m	1.10x	9.72 m	2.65x
Ours (CPU)	2.74 m	8.99x	2.54 m	6.38x	2.64 m	2.12x	2.72 m	9.46x
	2/3 Satin		1/4 Satin		Shot Silk A		Shot Silk B	
	Time	Speedup	Time	Speedup	Time	Speedup	Time	Speedup
PT	8.29 m	1.00x	19.48 m	1.00x	11.11 m	1.00x	8.17 m	1.00x
DS	7.08 m	1.17x	10.46 m	1.86x	8.11 m	1.37x	6.32 m	1.29x
MFT*	6.12 m	1.35x	9.47 m	2.06x	6.33 m	1.75x	6.16 m	1.33x
Ours (CPU)	2.70 m	3.07x	2.81 m	6.93x	2.67 m	4.16x	2.71 m	3.02x

	Fleece	Gabardine	Silk	4/1 Satin	2/3 Satin	1/4 Satin	Shot Silk A	Shot Silk B
Ours (GPU) time	25.57 s	13.98 s	14.99 s	18.86 s	18.86 s	22.47 s	29.26 s	25.52 s

🟥 Smaller values are better 🟩 Larger values are better

In addition to path tracing (PT), dual scattering (DS), the CPU version of our algorithm, and the GPU version, we also provide a conservative estimate of the time used by MFT. (*MFT's running time is estimated by running path tracing up to 6 accurate scattering events. We do not estimate the time MFT needs for photon tracing and stochastic matrix inversion and simply set it to 0.) The CPU-based algorithms ran on a 192-core cluster while the GPU version of our algorithm ran on a single GPU. We separate out the timings of the GPU algorithm since it uses different hardware from the rest. Note that rendering is very fast, taking from 14 seconds to 30 seconds on a single GPU compared to minutes on 192 CPU cores.

Kai Schröder, Reinhard Klein, and Arno Zinke. 2011. A Volumetric Approach to Predictive Rendering of Fabrics. In *Proceedings of the Twenty-second Eurographics Conference on Rendering (EGSR '11)*. Eurographics Association, 1277–1286. https://doi.org/10.1111/j.1467-8659.2011.01987.x

Kai Schröder, Arno Zinke, and Reinhard Klein. 2014. Image-Based Reverse Engineering and Visual Prototyping of Woven Cloth. *IEEE Transactions on Visualization and Computer Graphics* PP, 99 (2014). https://doi.org/10.1109/TVCG.2014.2339831

Li Shen, Jieqing Feng, and Baoguang Yang. 2013. Exponential Soft Shadow Mapping. *Computer Graphics Forum* (2013). https://doi.org/10.1111/cgf.12156

Peter-Pike Sloan, Jesse Hall, John Hart, and John Snyder. 2003. Clustered Principal Components for Precomputed Radiance Transfer. *ACM Trans. Graph.* 22, 3 (July 2003), 382–391. https://doi.org/10.1145/882262.882281

Peter-Pike Sloan, Jan Kautz, and John Snyder. 2002. Precomputed Radiance Transfer for Real-time Rendering in Dynamic, Low-frequency Lighting Environments. *ACM Trans. Graph.* 21, 3 (July 2002), 527–536. https://doi.org/10.1145/566654.566612

Peter-Pike Sloan, Ben Luna, and John Snyder. 2005. Local, Deformable Precomputed Radiance Transfer. *ACM Trans. Graph.* 24, 3 (July 2005), 1216–1224. https://doi.org/10.1145/1073204.1073335

Yu-Ting Tsai and Zen-Chung Shih. 2006. All-frequency Precomputed Radiance Transfer Using Spherical Radial Basis Functions and Clustered Tensor Approximation. *ACM Trans. Graph.* 25, 3 (July 2006), 967–976. https://doi.org/10.1145/1141911.1141981

Jiaping Wang, Peiran Ren, Minmin Gong, John Snyder, and Baining Guo. 2009. All-frequency Rendering of Dynamic, Spatially-varying Reflectance. In *ACM SIGGRAPH Asia 2009 Papers (SIGGRAPH Asia '09)*. ACM, New York, NY, USA, Article 133, 10 pages. https://doi.org/10.1145/1661412.1618479

Rui Wang, John Tran, and David Luebke. 2006. All-frequency Relighting of Glossy Objects. *ACM Trans. Graph.* 25, 2 (April 2006), 293–318. https://doi.org/10.1145/1138450.1138456

Kun Xu, Li-Qian Ma, Bo Ren, Rui Wang, and Shi-Min Hu. 2011. Interactive Hair Rendering and Appearance Editing Under Environment Lighting. In *Proceedings of the 2011 SIGGRAPH Asia Conference (SA '11)*. ACM, New York, NY, USA, Article 173, 10 pages. https://doi.org/10.1145/2024156.2024207

Kun Xu, Wei-Lun Sun, Zhao Dong, Dan-Yong Zhao, Run-Dong Wu, and Shi-Min Hu. 2013. Anisotropic Spherical Gaussians. *ACM Trans. Graph.* 32, 6, Article 209 (Nov. 2013), 11 pages. https://doi.org/10.1145/2508363.2508386

Baoguang Yang, Zhao Dong, Jieqing Feng, Hans-Peter Seidel, and Jan Kautz. 2010. Variance Soft Shadow Mapping. *Computer Graphics Forum* 29, 7 (2010), 2127–2134. https://doi.org/10.1111/j.1467-8659.2010.01800.x

Shuang Zhao, Miloš Hašan, Ravi Ramamoorthi, and Kavita Bala. 2013. Modular Flux Transfer: Efficient Rendering of High-resolution Volumes with Repeated Structures. *ACM Trans. Graph.* 32, 4, Article 131 (July 2013), 12 pages. https://doi.org/10.1145/2461912.2461938

Shuang Zhao, Wenzel Jakob, Steve Marschner, and Kavita Bala. 2011. Building Volumetric Appearance Models of Fabric Using Micro CT Imaging. In *ACM SIGGRAPH 2011 Papers (SIGGRAPH '11)*. ACM, New York, NY, USA, Article 44, 10 pages. https://doi.org/10.1145/1964921.1964939

Shuang Zhao, Wenzel Jakob, Steve Marschner, and Kavita Bala. 2012. Structure-aware Synthesis for Predictive Woven Fabric Appearance. *ACM Trans. Graph.* 31, 4, Article 75 (July 2012), 10 pages. https://doi.org/10.1145/2185520.2185571

Kun Zhou, Hujun Bao, Wei Hua, Zhong Ren, Weifeng Chen, Minghao Pan, and Rui Wang. 2013. Analytic Double Product Integrals for All-Frequency Relighting. *IEEE Transactions on Visualization and Computer Graphics* 19, undefined (2013), 1133–1142. https://doi.org/doi.ieeecomputersociety.org/10.1109/TVCG.2012.152

Arno Zinke and Andreas Weber. 2007. Light Scattering from Filaments. *IEEE Transactions on Visualization and Computer Graphics* 13, 2 (2007), 342–356. https://doi.org/10.1109/TVCG.2007.43

Arno Zinke, Cem Yuksel, Andreas Weber, and John Keyser. 2008. Dual Scattering Approximation for Fast Multiple Scattering in Hair. *ACM Trans. Graph.* 27, 3, Article 32 (Aug. 2008), 10 pages. https://doi.org/10.1145/1360612.1360631