


Approximations for the distribution of microflake normals

Nelson Max^{1,2}  · Tom Duff¹ · Ben Mildenhall³ · Yajie Yan⁴

© Springer-Verlag Berlin Heidelberg 2017

Abstract Scenes in computer animation can have extreme complexity, especially when high resolution objects are placed in the distance and occupy only a few pixels. A useful technique for level of detail in these cases is to use a sparse voxel octree containing both hard surfaces and a participating medium consisting of microflakes. In this paper, we discuss three different methods for approximating the distribution of normals of the microflakes, which is needed to compute extinction, inscattering of attenuated direct illumination, and multiple scattering in the participating medium. Specifically, we consider (a) k means approximation with k weighted representatives, (b) expansion in spherical harmonics, and (c) the distribution of the normals of a specific ellipsoid. We compare their image quality, data size, and computation time.

Keywords Microflake · Sparse voxel octree · Volume rendering · Distribution of normals

Electronic supplementary material The online version of this article (doi:[10.1007/s00371-017-1352-2](https://doi.org/10.1007/s00371-017-1352-2)) contains supplementary material, which is available to authorized users.

✉ Nelson Max
max@cs.ucdavis.edu

Tom Duff
td@pixar.com

Ben Mildenhall
bmild@berkeley.edu

Yajie Yan
yajieyan@wustl.edu

- ¹ Pixar Animation Studios, Emeryville, CA, USA
- ² Present Address: University of California, Davis, CA, USA
- ³ University of California, Berkeley, CA, USA
- ⁴ Washington University in St. Louis, St. Louis, MO, USA

1 Introduction

Scene models for computer animation such as in Fig. 1 often have extreme complexity if the original detailed models are used for objects in the distance. One method of dealing with this data explosion is to have multiple levels of detail, so that simplified models can be used for more distant objects. Another method is to summarize into a volume data structure the aspects of the model that affect rendering. We are using the latter method, with a sparse voxel octree (SVO), of the sort described by [9]. The polygons (or micropolygons, see [2]) are scan converted into the leaves of the octree structure, averaging the plane equation, plane normal variance, specular and diffuse color, and other shading quantities such as surface roughness. These quantities are then averaged further into the higher octree levels. When marching along a viewing ray through the octree, cells are selected at the octree level appropriate to the ray differential (randomly jittered to break up any visible transitions in shading from different SVO levels), and the ray is intersected with the cell's stored average surface plane, as clipped to the cell volume. The first point intersected is shaded by combining the plane normal variance into the surface roughness. We refer to this as the “hard surface” method.

One of our goals was to model distant vegetation, as in Fig. 1. The hard surface method does not work well for this, because we are intersecting the ray with the average plane, while the octree cell should actually represent many small plant leaves inside its volume. In addition, even a single tiny leaf in the volume may cause an intersection with its extended plane, making the apparent size of the leaf bloom larger. To solve these problems, we treat the leaf surfaces in an octree cell as a volume density of microflakes (infinitesimal pieces of flat surface) and use volume rendering methods to composite the inscattering and extinction along the ray as

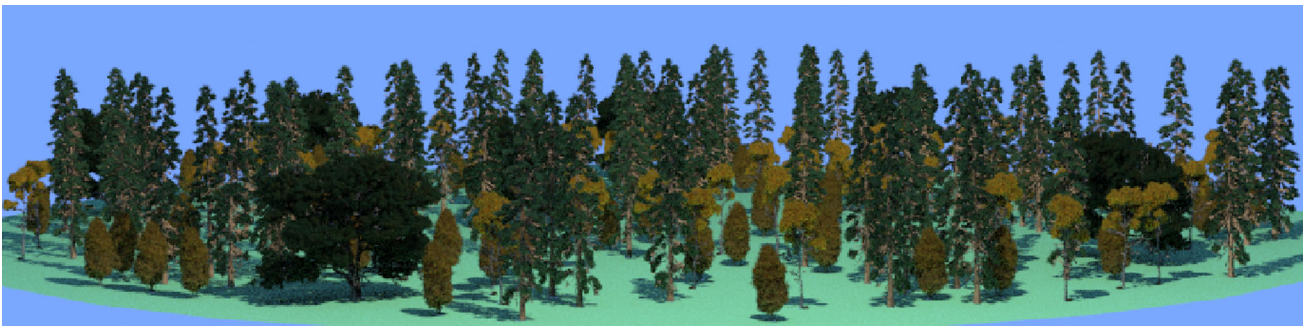


Fig. 1 A wide angle view of a forest of 218 trees of various species, path traced using the spherical harmonic microflake normal distribution. The sparse voxel octree used 2,434,796,669 bytes, while a single pine tree polygon representation needed 1,816,829,593 bytes

it steps through the octree cells. Both the inscattering and the extinction depend on the distribution of the microflake surface normals. This paper compares several methods of representing this distribution by approximations specified by a small number of parameters. Ideally, such approximations should give:

1. A compact representation,
2. Efficient averaging up the octree hierarchy,
3. Efficient computation of the extinction coefficient,
4. Efficient computation of the inscattering,
5. Efficient sampling of the scattered ray direction for global illumination path tracing, and
6. Final renderings that well approximate converged ray traced or path traced images of the original geometry.

The contributions of this paper are:

- Two new methods to approximate a distribution of normals, by k means representatives and by spherical harmonics (SH), and
- Their comparison with each other and the SGGX distribution of Heitz et al. [8], which uses the distribution of normals of a specific ellipsoid.

Section 2 discusses related work, and Sect. 3 describes the three approximations and how their fitting parameters are computed. Section 4 discusses how the extinction and inscattering are computed from the distribution of microflake normals. Section 5 shows how sample directions for path tracing are computed, Sect. 6 discusses optimizations for double-sided microflakes, and Sect. 7 describes compression of the data. Section 8 presents image accuracy, data size, and timing results, which are discussed in Sect. 9. The appendix has mathematical details for the SH approximation.

2 Related work

The fundamental paper on level of detail was Crow [4], which first suggested designing several different polygonal models for the same object. This idea was refined by Hoppe to create the simplified models automatically [11], and continuously transition between them [12].

Octrees have long been used to describe sparse volume data and speed up ray tracing. Neyret [23] used a hierarchical octree volume with the reflectance in each cell represented by the distribution of normals of an ellipsoid. Heitz et al. [8] did something similar in the context of the SVO of [9]. We have also used a version of this SVO, enhanced to store the various approximations to the distribution of microflake normals, and with some new compression mechanisms.

The basic principles of volume rendering participating media, accounting for the optical effects of extinction and inscattering, are explained in [20]. Normally, the extinction is isotropic, independent of the viewing direction. The scattering phase function, giving the distribution of scattering directions ω_o from an incoming direction ω_i , is also normally isotropic, in the sense that it depends only on the angle between ω_i and ω_o . However, in the case of an anisotropic distribution of microflake normals, these isotropies are not present, and the full dependence of the extinction and inscattering on both ω_o and ω_i must be accounted for. This involves the distribution of visible normals, which accounts for both masking and foreshortening for bumpy surfaces [6], and for foreshortening only in the case of microflakes, as discussed in [8], and briefly in Sect. 4 below. An early computer graphics paper considering the optical effects of a microflake normal distribution is [21], which simulated multiple scattering in a tree canopy. More recently, Jacob et al. [13] discussed the radiation transport equation for a general anisotropic distribution of microflake normals, and also the corresponding diffusion approximation. Zhao et al. [32, 33] modeled cloth as a collection of fibers extracted from micro-CT and modeled the microfacet normal distribution of a fiber as a gaussian in the dot product of the microflake normal and the fiber axis.

3 Representing the distribution of microflake normals

Let $g(x, \omega_n)$ be the local probability distribution of the microflake normals with direction ω_n at position x , let $\rho(x)$ be the density of total microflake area per unit volume, and

let $h(x, \omega_n) = \rho(x)g(x, \omega_n)$. Then, $h(x, \omega_n)d\omega_n$ is the total area per unit volume of microflakes with normals within a solid angle $d\omega_n$ about ω_n . We will usually mean this scaled $h(x, \omega_n)$ when we say the distribution of microflake normals below, even though this is not a probability distribution.

We will compare three different methods of approximating this distribution of microflake normals, by:

- (a) A finite number k of weighted δ -functions of unit direction,
- (b) An SH expansion of fixed maximum order L , or
- (c) the distribution of normals of an origin-centered ellipsoid.

If the degrees of freedom are all expressed as floats, then method (a) requires $4k$ floats, method (b) requires $(L + 1)^2$ floats, and method (c) requires only six floats at each SVO node. Later we will discuss compression of these data.

When scan converting polygons into the SVO, we recursively split the polygons by the slicing planes defining the SVO volume cells, until the fragments are the size of the leaf cells. For the hard surface contribution, we compute the area inside each intersected leaf cell and add that area and also the area-weighted diffuse color, specular color, unit normal, and other shading parameters like surface roughness, to the accumulated values for that leaf cell. After all the polygons are scan converted, we add these weighted quantities to the parent cells up the octree hierarchy and then divide all these summed quantities by the sum of the weights. We also compute the variance of the normals. An SVO cell can contain both hard surfaces and microflakes, and the colors of the microflakes are averaged separately into the SVO as above, so that the plant leaves can have a different color than the hard surface trunks and branches. The calculations to fit each of the three microflake normal distribution approximation methods into this SVO framework are described next.

The k means algorithm for method (a) converges to a local minimum of its cost function, which depends on the initial cluster center normal representatives chosen. To get a good spread in the initial representatives, a first pass through the polygon fragments finds up to k representative normals per octree cell. We take a new area-weighted representative normal whenever there are less than k and the current polygon fragment normal differs from all existing representative normals by more than a threshold angle. Otherwise we add the area-weighted fragment normal to the representative to which is closest in direction. Then, in subsequent passes through the polygons, we reassign each polygon fragment's contribution to the representative to which it is closest and compute new weighted average representative directions, thus implementing a weighted k means algorithm. When all polygons are handled, we use the same procedure to go up the octree hierarchy, assigning child cell weighted representatives to parent

cell representatives, and computing the area-weighted average of the normals in each representative's group. Currently, we use only two passes through all the polygons in computing the k means for the leaf cells of the SVO, since such passes are expensive, but more passes to propagate the leaf representatives into the k means of their ancestor cells up the hierarchy, since this does not need the full polygon data. The resulting approximation is an area-weighted sum of the δ -functions for the representative unit normals in each SVO cell. We also save the variance of the normals in the cluster assigned to each representative, estimated from the length of the weighted average normal using the method of [29], which does not require a separate pass through the polygon data. This variance per representative is the fourth of the $4k$ floats mentioned above. For greater accuracy, we could store the covariance matrix of the distribution of the microfacet normals assigned to each representative normal, as in [25], as done for method (c) below, but this would take 6 extra floats per representative normal, instead of one.

Method (b) fits the distribution of normals using the real spherical harmonic basis functions $Y_{lm}(\theta, \phi)$.

Since the SH basis functions are orthonormal, the best L_2 fit to a function $h(\theta, \phi)$, of the form

$$h(\theta, \phi) \approx \sum_{l=0}^L \sum_{m=-l}^l h_{lm} Y_{lm}(\theta, \phi) \tag{1}$$

has coefficients given by

$$h_{lm} = \int_0^{2\pi} \int_0^\pi Y_{lm}(\theta, \phi) h(\theta, \phi) \sin\theta \, d\theta \, d\phi. \tag{2}$$

So for an area-weighted δ -function representing the contribution of a polygon fragment with a constant unit normal to an SH basis function coefficient, this integral is just the area times that basis function evaluated at that normal. These contributions are added into the leaf cells and then into parent cells up the octree hierarchy. There is no need to divide by the sum of the weights, because the goal is to approximate an area-weighted density $h(x, \omega_n)$. However, since this is an area per unit volume, the accumulated sum must be divided by the volume of the octree cell. [This is also the case for method (a).] We have experimented with spherical harmonic orders L up to 4.

For method (c), Heitz et al. [8] define the SGGX distribution of squared projected areas of an ellipsoid as a quadratic form in the unit projection direction vector. They define it in the coordinate system of the unit eigenvector directions (the ellipsoid's major, semi-major, and minor axis directions) by a diagonal matrix of the squared projected areas in these directions and use the rotation matrix formed from these unit eigenvector directions to transform this quadratic form into the world coordinate system. They prove that the square root

of this quadratic form evaluated at any other direction gives the projected area of the ellipsoid in that direction. For a polygon fragment from a smooth surface, they use a “surface-like” ellipsoid with unit projected area in the surface normal direction, and very small projected area in the perpendicular directions, while for a fragment of a curve (for fur, hair, or pine needles), they use a “fiber-like” ellipsoid, with a small projected area along the curve tangent direction, and unit projected areas along perpendicular directions.

For the images in that paper, they just accumulate the six independent elements of the positive definite 3 by 3 symmetric matrix for the quadratic form into the leaf cells and then into the octree hierarchy, but they admit that this does not properly accumulate the distributions of normals of child cells into parents. So we use the two pass method they describe in their Sect. 4.3, subsection “Parameter Estimation from Arbitrary Distributions.” In the first pass, we accumulate the covariance matrices for the distributions. The covariance matrix contribution to an octree leaf cell of a clipped polygon fragment with unit normal column vector ω is simply the fragment’s area times the outer product matrix $\omega\omega^T$. These contributions are summed in each leaf cell and then up the hierarchy. At the end of this pass, we compute and save the eigenvectors of the resulting symmetric covariance matrix in each octree cell. In a second pass through the polygons, we compute the projected areas of the polygon fragments in these eigenvector directions, not just for the leaf cells, but for all the octree cells encountered in the recursive polygon slicing. Then, we use the square of these summed projected areas to get the diagonal matrix for the quadratic form in the eigenvector coordinate system and transform it into the world coordinate system using the matrix of unit eigenvectors.

4 Computing extinction and inscattering

The microflake area per unit volume near a position x , as projected in the viewing direction ω_o , is

$$\sigma_t(x, \omega_o) = \int_{\Omega} h(x, \omega_n) \langle \omega_n, \omega_o \rangle d\omega_n, \tag{3}$$

where Ω is the unit sphere, and the foreshortening projection factor $\langle -, - \rangle$ represents the nonnegatively clamped dot product if one-sided flakes are being used, or the absolute value of the dot product if double-sided flakes are being used, and ω_o is actually the negative of the viewing ray direction, as in the usual convention for representing BRDFs. According to the reasoning in [20], this $\sigma_t(x, \omega_o)$ is the extinction coefficient, or volumetric attenuation coefficient, expressing the fraction of flux intercepted by the flakes per (infinitesimal) unit length. Figure 2 (left) shows a slab of front face area 1 and

infinitesimal thickness ds , and thus with volume ds , a sample of microflakes from the distribution of normals $h(x, \omega_n)$, and their projections on the face of the slab closest to the viewer. The slab thickness ds is assumed to be so small that the chance that the microflake projections overlap each other approaches zero. Therefore, sum of their projected areas is the slab’s opacity, expressed by multiplying Eq. (3) by the slab volume ds to convert area per unit volume to area in the slab. So the slab’s transparency is $1 - \sigma_t(x, \omega_o)ds$.

To compute the inscattering, let $f(\omega_i, \omega_o; \omega_n)$ be the BRDF of a microflake with normal ω_n . Accounting for both the foreshortening in the incident direction ω_i (for irradiance) and in the viewing direction ω_o (for viewing visibility), the inscattering per unit length along the viewing ray is

$$S(x, \omega_o) = \int_{\Omega} \int_{\Omega} h(\omega_n) f(\omega_i, \omega_o; \omega_n) \langle \omega_n, \omega_i \rangle \langle \omega_n, \omega_o \rangle L(x, \omega_i) d\omega_n d\omega_i, \tag{4}$$

where $L(x, \omega_i)$ is the incoming radiance at x coming from direction ω_i . See the right half of Fig. 2, where the colors represent the microflakes’ reflected radiance in direction ω_o .

The quantities $\sigma_t(x, \omega_o)$ and $S(x, \omega_o)$ are used to compute the volume rendering integral along the viewing ray in direction $-\omega_o$, by integrating this inscattering contribution along the ray, as attenuated by the transparency between each slab and the viewpoint E :

$$L(E, \omega_o) = \int_0^D \exp\left(-\int_0^s (\sigma_t(x(t), \omega_o) dt)\right) S(x(s), \omega_o) ds, \tag{5}$$

where $x(t) = E - t\omega_o$ is the arclength parametrization of the ray leaving the viewpoint E in direction $-\omega_o$, and D is the parameter at which this ray hits a hard surface or exits the SVO data volume. See [20] for a detailed derivation. As shown by the equations above, the extinction and inscattering depend on the distribution of *visible* normals, due to the foreshortening factor $\langle \omega_n, \omega_i \rangle$. However, the full geometric attenuation factor accounting for shadowing and masking when considering surface microfacets is not present here, because these effects are instead included in the volume attenuation calculations in Eq. (5) above, and in computing similarly how much light from a distant source reaches x and contributes to $L(x, \omega_i)$ in Eq. (4).

As is usual in volume rendering, we numerically compute the integral in equation (5) by marching along the viewing ray through the octree, determining the successive segments in which it intersects cells at the level locally appropriate to its ray differential, and incrementally accumulating the transparency

Fig. 2 *Left* extinction in a slab of area 1 and infinitesimal thickness ds , *Right* inscattering in that slab

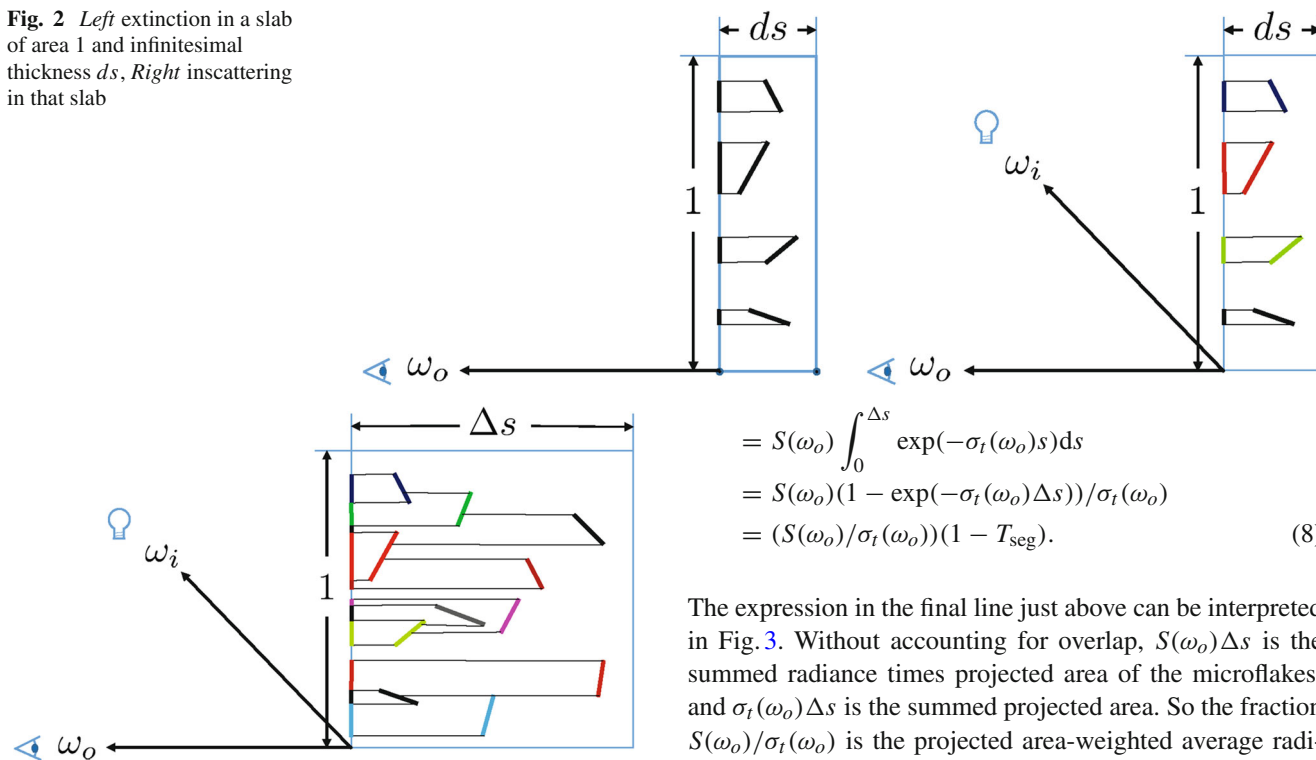


Fig. 3 Mutually occluding microflakes in a slab of finite thickness Δs

$$T(s) = \exp\left(-\int_0^s (\sigma_t(x(t), \omega_o) dt)\right) \tag{6}$$

and also the ray segment's contribution to the radiance arriving at E in Eq. (5).

We assume that $h(x, \omega_n)$, and therefore $\sigma(x, \omega_o)$ and $S(x, \omega_o)$, are constant in each cell, so we leave the location x out of the equations below. If the ray segment has length Δs , the update for T should not just be $T = T \times (1 - \sigma_t \Delta s)$, accounting for the unoccluded fraction of the slab face in Fig. 2, because Δs is not infinitesimal, and the projected microflakes can overlap, as shown in Fig. 3. It is more accurate to compute the segment transparency integral

$$T_{\text{seg}} = \exp\left(-\int_0^{\Delta s} (\sigma_t(\omega_o) dt)\right) = \exp(-\sigma_t(\omega_o) \Delta s) \tag{7}$$

which accounts for this overlap. Then, we use the update $T = T \times T_{\text{seg}}$. Similarly, instead of incrementing $L(E, \omega_o)$ by $T \times S(x, \omega_o) \times \Delta s$ as is sometimes done in volume rendering, it is more accurate to compute the inscattering integral from the segment analytically, as

$$L_{\text{seg}} = \int_0^{\Delta s} \exp\left(-\int_0^s (\sigma_t(\omega_o) dt)\right) S(\omega_o) ds$$

$$\begin{aligned} &= S(\omega_o) \int_0^{\Delta s} \exp(-\sigma_t(\omega_o)s) ds \\ &= S(\omega_o) (1 - \exp(-\sigma_t(\omega_o)\Delta s)) / \sigma_t(\omega_o) \\ &= (S(\omega_o) / \sigma_t(\omega_o)) (1 - T_{\text{seg}}). \end{aligned} \tag{8}$$

The expression in the final line just above can be interpreted in Fig. 3. Without accounting for overlap, $S(\omega_o)\Delta s$ is the summed radiance times projected area of the microflakes, and $\sigma_t(\omega_o)\Delta s$ is the summed projected area. So the fraction $S(\omega_o)/\sigma_t(\omega_o)$ is the projected area-weighted average radiance of the microflakes. Since the depth order for occlusion is random, this is also the average radiance of the colored segments shown in Fig. 3 on the slab face. The area *not* covered by these segments is T_{seg} , so the area that *is* covered is $1 - T_{\text{seg}}$. Thus, the inscattered radiance L_{seg} in equation (8) is the average radiance times the area covered. The update to $L(E, \omega_o)$ in equation (5) to include this segment's contribution is $L(E, \omega_o) = L(E, \omega_o) + T \times L_{\text{seg}}$ where T is the accumulated transparency of all the previous segments, before being updated to include the current one.

Now we discuss how to compute $\sigma_t(x, \omega_o)$ and $S(x, \omega_o)$ for each of the three methods for approximating $h(x, \omega)$ in a cell. For method (a), the distribution of normals is a sum of weighted δ -functions, with weights c_j , in directions ω_j :

$$h(x, \omega) = \sum_{j=1}^k c_j \delta(\omega - \omega_j). \tag{9}$$

Substituting into equation (3), the extinction is

$$\sigma_t(x, \omega_o) = \sum_{j=1}^k c_j \langle \omega_j, \omega_o \rangle. \tag{10}$$

For a perfectly diffuse surface with albedo α and thus diffuse BRDF α/π , and a small light source in direction ω_i whose radiance times subtended solid angle at x is I , equation (4) for the inscattering gives

$$S(x, \omega_o) = \frac{I\alpha}{\pi} \sum_{j=1}^k c_j \langle \omega_j, \omega_o \rangle \langle \omega_j, \omega_i \rangle. \tag{11}$$

Specular reflection can be handled similarly, using the variance of the polygon fragment normals clustered with each of the k representative normals, calculated when constructing the octree, to determine the width and maximum radiance of each specular peak.

For method (b), the integral for the extinction coefficient in equation (3) of the SH expansion for the microflake normal density times the clamped cosine can be computed as a weighted sum of the spherical harmonic basis functions evaluated at the viewing direction, according to Sect. 4 of [26] [See Eq. (15) in the “Appendix” below]. If the two clamped cosines in the integral for the inscattering in Eq. (4) are both expanded in spherical harmonics as in [26], this integral can be expressed as a weighed sum of triple product integrals of spherical harmonic basis functions [See the “Appendix”, equation (16)]. Most of these triple product integrals are zero for the higher orders, and we precomputed the rest and stored them in a sparse array. Jacob et al. [13] also discussed the above method of computing the extinction coefficient from a spherical harmonic representation, but not how to compute the inscattering integral.

For microfacet-generated mirror reflection, we need to evaluate the spherical harmonic visible microfacet normal density approximation $\langle \omega, \omega_o \rangle h(\omega)$ at the half angle direction $\omega = \omega_h = (\omega_i + \omega_o) / |\omega_i + \omega_o|$ that specifies the microfacet normal which reflects light from direction ω_i into direction ω_o , as in [30] and [3]. We must multiply by the change of variables factor $d\omega_h/d\omega_o = 1/(4\langle \omega_h, \omega_i \rangle)$ as given in [27] and explained in the introduction to Chap. 7 of [16] and briefly in [31]. Since $\langle \omega_h, \omega_o \rangle = \langle \omega_h, \omega_i \rangle$ these two factors cancel.

For method (c), we use the routines in [7]. For the projected microflake area in the viewing direction in equation (3), we take the square root of the quadratic form of squared projected areas, evaluated at the viewing direction. The perfect mirror microflake specular reflection can be found from the microflake normal distribution evaluated at the half angle direction ω_h as in method (b) above. This distribution is given by equations (24), (11), and (12) of [8]. The diffuse reflection is more difficult, because, unlike methods (a) and (b), an analytic expression for the integral in Eq. (4) is not known, even for a point or directional light source. So as proposed in [8], we instead sample the distribution of visible normals (see the next section) and use the diffuse BRDF for a flat surface with the resulting normal. This technique introduces noise, however, and is only useful in a context where multiple ray samples per pixel are already being used for other purposes like anti-aliasing, depth of field, or global illumination. We may also need to use it for the spherical harmonics method (b), if there are double-sided microflakes. (See Sect. 6 below.)

5 Computing sample directions for global illumination

To efficiently compute inscattered illumination from a whole environment sphere Ω in equation (4), or the next bounce direction in path tracing methods for global illumination by Monte Carlo integration, once the Monte Carlo volume rendering algorithm determines that volume scattering should take place (see [15]), we need to importance-sample the distribution of scattering directions ω_i for a viewing ray or continuing path direction $-\omega_o$. This involves first sampling a normal from the distribution of visible microfacet normals $\langle \omega_o, \omega_n \rangle h(x, \omega_n)$, or rather the normalized probability distribution proportional to it, and then sampling the BRDF for a flat microflake surface with this normal.

For method (a), we generate a random number uniformly distributed in the interval $[0, 1]$, use it with the inverse of the cumulative distribution of the visible normal probabilities $\langle \omega_j, \omega_o \rangle c_j / \sum \langle \omega_j, \omega_o \rangle c_j$ to chose surface j with probability proportional to $\langle \omega_j, \omega_o \rangle c_j$, and then sample the BRDF for a surface with normal ω_j .

For method (b), we show in equation (18) of the “Appendix” that the importance of the incoming radiance from the next diffuse bounce direction also has an SH representation, with coefficients computed from ω_o and the SH coefficients of the microflake normal distribution. Thus, we can use the algorithm of [14] for sampling spherical harmonics distributions. It divides the (ϕ, θ) range of the unit sphere iteratively into quadrants as in a quadtree, integrates the distribution over each quadrant (using precomputed tables), and uses a random number for the inverse cumulative distribution method to choose a quadrant at each stage of the iteration.

One problem with this sampling algorithm is that the best spherical harmonic fit of a nonnegative function on the unit sphere can have some negative values, due to ringing. In our implementation, quadrants with negative integrals are given probability zero, and the probabilities of the remaining quadrants are normalized to add up to one. However, when negative lobes of the SH function cause distortions in the distribution of our samples by inappropriately decreasing the probability of sampling the positive lobes in a quadrant, we resort to rejection sampling, precomputing and saving in each octree cell the maximum of its spherical harmonic fitting function. The rejection test rejects samples with negative function values and produces directions with probability proportional to positive function values.

For method (c), we use the sampling algorithm in Sect. 1.6 of [7].

6 Double-sided microflakes

In our application, the vegetation leaf polygons are shaded on both sides with the same color, as is the case for the microflakes in [13]. For method (a), we can take advantage of this and use a smaller number k of representatives of the microflake normals, by comparing both a new normal n and its negative $-n$ with all the representatives when choosing the closest. Then, when shading, we can use either the representative normal, or its negative, whichever is front-facing.

For method (b), we can conceptually use the δ -functions with both n and $-n$ when computing the spherical harmonic representation, as in Sect. 3. The basis functions $Y_{lm}(n)$ are even functions of the unit vector n , i.e., $Y_{lm}(-n) = Y_{lm}(n)$, when l is even, and odd functions of n , i.e., $Y_{lm}(-n) = -Y_{lm}(n)$, when l is odd. Thus, when summing the contributions from n and $-n$, all terms with odd l disappear, and only $(\lfloor L/2 \rfloor + 1)(2\lfloor L/2 \rfloor + 1)$ terms remain, which for large L is about half of the total $(L+1)^2$. For order $L = 4$, there remain only 15 terms instead of 25. This saves both space and computation time. However, when Eq. (16) in the “Appendix” is applied to the shading of a double-sided smooth surface we get an effect like double-sided lighting (as if from direction $-\omega_i$ as well as ω_i , see Fig. 10), because the reversed normal also contributes to the shading. To avoid this, we can shade by sampling the visible normal distribution, reversing the resulting normal if it has a negative dot product with ω_0 , and then applying the BRDF for a surface with this normal. This normal sampling introduces noise, as with the shading for the ellipsoid method. An alternative is to keep all $(L+1)^2$ terms, and apply Eq. (16) twice, using the single-sided clamped version of the $\langle \omega_n, \omega_i \rangle$ and $\langle \omega_n, \omega_o \rangle$ factors, and for the second time reversing both the viewing and lighting directions, which is equivalent to reversing the normal. On the other hand, the double-sided lighting may actually be appropriate for vegetation leaves, if diffuse transmission is considered, as well as diffuse reflection.

Method (c) naturally represents double-sided microflakes, since the ellipsoids are centrally symmetric and reflect from all sides.

7 Compression

In order to reduce the size of our SVO, we compress the data inside each cell after octree construction and dynamically decompress the requested values as rendering occurs. We use the Best Fit Normal [19] method to compress normals, and a dictionary-based method to compress other related floating point numbers.

Best Fit Normal compresses a 3-float normal (12 bytes) to 3 bytes, achieving a compression ratio of 4. Essentially, this method assigns a normal to the best fit voxel’s index within a

voxel space of size 256^3 . The trick is how to find the best fitting voxel for each unit normal. To do this, instead of treating the normal as a unit vector, we treat it as a ray with arbitrary length. We start this ray at the center of the 256^3 voxel space and then walk along it. For every voxel intersected by the ray, we compute the distance from that voxel center to the ray and assign the index of the closest voxel to the normal as its compressed form. The decompression of such a form amounts to a subtraction and a normalization.

Our dictionary-based compression scheme takes in a set of floats, builds a dictionary, and assigns each float to its closest entry in the dictionary. To build the dictionary, we use an approximate k means method [18] for fast clustering, and extract N entries as specified by the user. For a single float, the compression ratio is then $32/\lceil \log N \rceil$ bits. Decompression is merely a lookup in the dictionary. We use this method for color components, and normal variance.

For compressing the 6 coefficients of the quadratic form in method (c), we use the suggestion of [8]. We take the square root of the absolute value, give it the sign of the original coefficient, and then convert it to a 16 bit half float [17]. For method (a), we specify each weighted representative normal as a scalar weight multiplying a unit normal vector, and use the Best Fit Normal method above for the unit normals. We use the half float of the square root for the scalar weights and the variances, and also for the spherical harmonics coefficients for method (b).

8 Results

We tested the methods described above on several tree models, including a broad-leafed cottonwood tree and a needle-leafed lodgepole pine. For the cottonwood “ground truth” images, we ray traced the full polygonal model at 8 by 8 supersampling resolution. We also generated images by ray tracing in the SVO, using the hard surface representation for the leaves, as well as for the trunks and branches. Then, we generated octrees and images using the hard surface representation for the trunk and branches, and ray tracing through the microflake volume density for the leaves, using the three approximations methods discussed above for the distribution of microflake normals. The octrees had resolution 1024^3 . We tested the spherical harmonic method for both order $L = 2$ with 9 basis functions and for order $L = 4$, with 25 basis functions, and tested all methods at both 8 by 8 supersampling (with a correspondingly reduced ray differential), and without supersampling, where the summarization qualities of the SVO are more important. Figure 4 shows the cottonwood tree in 640 by 640 pixel resolution images with 8 by 8 supersampling, which used the lowest level leaf cells of the octree when ray tracing. Note that the shading is washed out for the $L = 2$ spherical harmonics, because the distribution of nor-

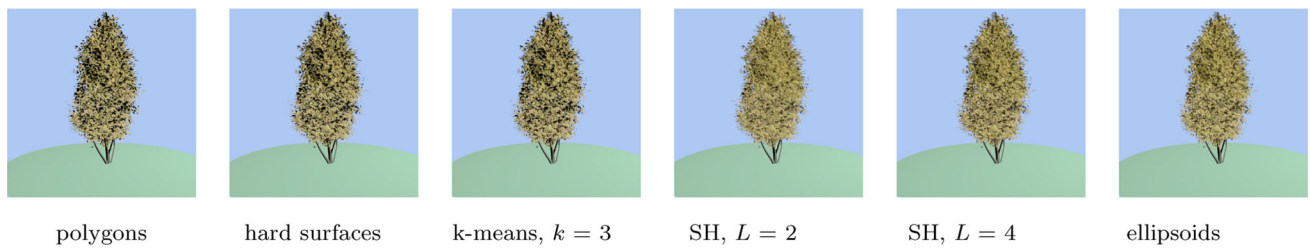


Fig. 4 Cottonwood tree, at 640 by 640 resolution, and 8 by 8 supersampling. From *left to right* ground truth image from original polygons, hard surface image from SVO, k means with $k = 3$, SH with $L = 2$, SH with $L = 4$, and ellipsoidal microflakes

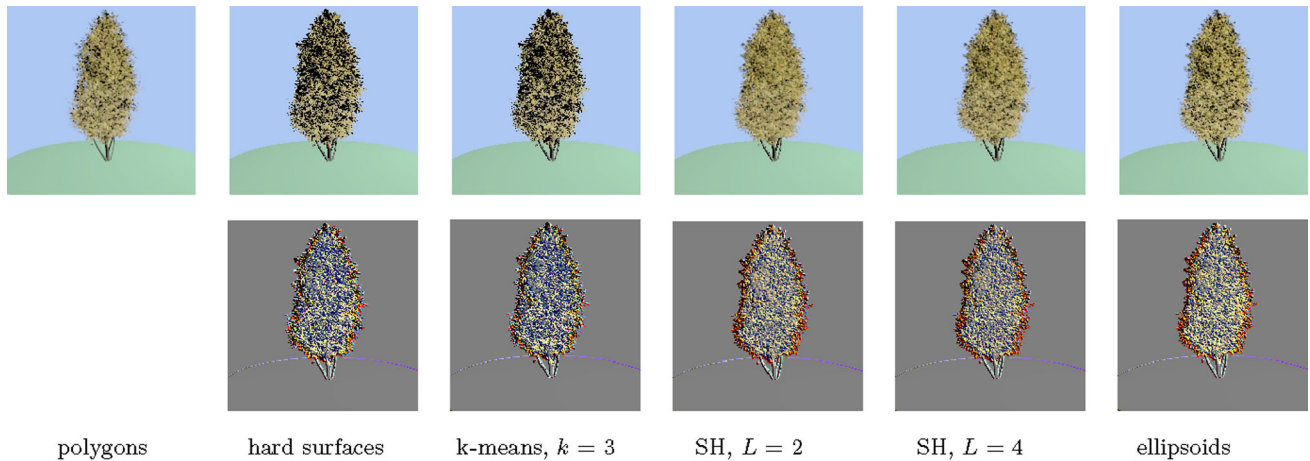


Fig. 5 Cottonwood tree, at 200 by 200 resolution, and no supersampling except for the ground truth image which is 8 by 8 supersampled. *Top row from left to right* ground truth image from original polygons, hard surface image from SVO, k means with $k = 3$, SH with $L = 2$,

SH with $L = 4$, and ellipsoidal microflakes. *Bottom row* differences of the rightmost five images in the *top row* and the leftmost ground truth image, magnified by a factor of five, with neutral gray representing zero

mals is too band limited. Figure 5 shows non-supersampled images at 200 by 200 resolution, on the same octrees as Fig. 4, where the much larger ray differentials cause the higher level internal nodes of the octree to be used. Both figures are without any shadows. The second row in Fig. 5 shows the differences between the SVO-based images in the first row and the ground truth ray traced and supersampled polygon-based image of the same resolution, multiplied by five.

For the lodgepole pine, we expanded the vertices of the polylines defining the curved pine needles into regular hexagons, so that the needle segments were represented by approximate hexagonal prisms. This means that the distribution of the normals to the six prism faces for a single needle segment can be exactly represented by the k means approximation with $k = 3$, when vectors of exactly opposite directions are considered equivalent. The octrees had resolution 512^3 . Figure 6 shows 200 by 400 resolution images. At the left is a ray traced image of the polygonal model, at 40 by 40 supersampling, and the subsequent images are from the SVO, using the different microflake normal distribution approximations, with 2 by 2 supersampling. The second row again shows differences.

Tables 1, 2, and 3 show the performance statistics for generating the images in Figs. 4, 5, and 6, respectively. The SVO sizes are in bytes, and the RMS errors are for the 3 floating point color components in the range from 0 to 1, including the sky and ground regions with no error. Table 2 used the same octrees as Table 1. The timings were done on an HP Z800 workstation with 12 Intel Xeon X5660 2.88GHz cores. The octree construction was serial, but the ray tracing was done in parallel using 10 cores, with the other 2 for task management and system calls. For the cottonwood tree, the rendering times were per frame and do not include initialization and loading the precomputed octree, while for the lodgepole pine, these initialization and loading times are included.

Figure 7 shows the cottonwood tree with shadows at 400 by 400 resolution, with 4 by 4 supersampling, rendered from a 512^3 SVO using the k means method (a) with $k = 3$. The left-hand image used ray traced shadows and took 346.2s to render. For the right-hand image, we first propagated the illumination into the leaf cells of the SVO by tracing a grid of 2068 by 1462 rays from the light source through the octree, depositing in each intersected cell the ray segment length and the length-weighted attenuated flux it carried. After all

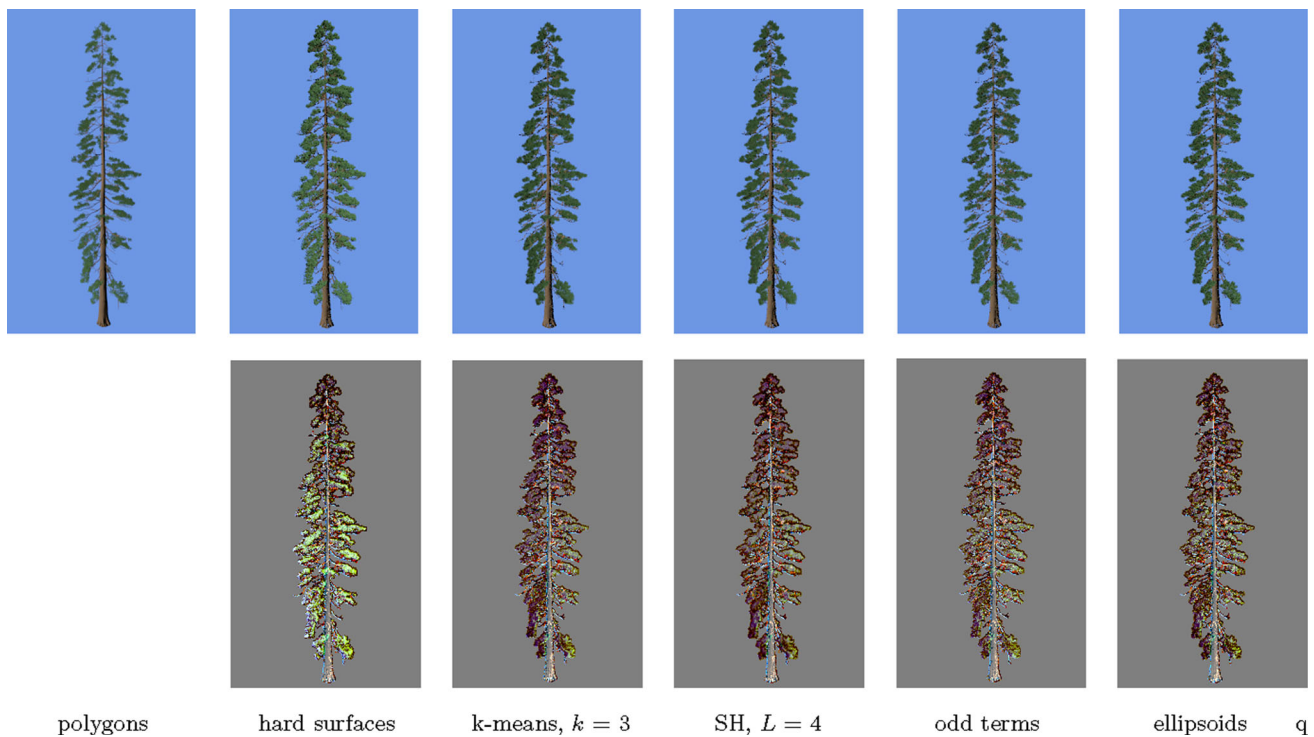


Fig. 6 Lodgepole pine tree, at 200 by 400 resolution. *Top row from left to right* ground truth image from original polygons; hard surface image from SVO; k means microflakes with $k = 3$; spherical harmonics microflakes with $L = 4$ with all 25 terms, and double-sided shading as in Sect. 6; spherical harmonics microflakes with $L = 4$ with only

the 15 odd order terms, shaded by sampling from the microflake normal distribution; and ellipsoidal microflakes. *Bottom row* differences of the rightmost five images in the *top row* and the leftmost ground truth image, magnified by a factor of five

Table 1 Statistics for cottonwood tree images at 640 by 640 resolution, with 8 by 8 supersampling

Method	SVO size	RMS	SVO tm	Render
Hard surface	1,062,142,164	0.0449	7:24	17.87
k means	2,468,438,261	0.0450	3:44	20.05
SH: $L = 2$	2,012,479,768	0.0454	8:13	45.19
SH: $L = 4$	2,906,925,160	0.0442	8:34	146.64
Ellipsoidal	2,347,893,040	0.0451	9:57	30.07

The last two columns show SVO construction time, in minutes:seconds, and rendering time, in seconds

Table 2 Statistics for cottonwood tree at 200 by 200 resolution

Method	64 RMS	64 Render	1 RMS	1 Render
Hard surface	0.0449	1.805	0.0857	0.097
k means	0.0449	2.036	0.0858	0.096
SH: $L = 2$	0.0464	4.531	0.0640	0.133
SH: $L = 4$	0.0464	14.336	0.0633	0.313
Ellipsoidal	0.0464	3.150	0.0670	0.105

RMS errors with the ray traced polygonal model and rendering times in seconds are shown for 64 samples per pixel, and 1 sample per pixel

Table 3 Statistics for lodgepole pine at 200 by 400 resolution

Method	SVO size	RMS	SVO tm	Render
Hard surface	123,734,707	0.067	13:54	0.48
k means	190,884,125	0.056	16:54	1.16
SH: 25 terms	201,329,829	0.050	15:49	1.58
SH: 15 terms	180,358,269	0.062	16:04	1.42
Ellipsoidal	178,261,368	0.057	34:29	1.97

The last two columns show SVO construction time, in minutes:seconds, and rendering time, in seconds

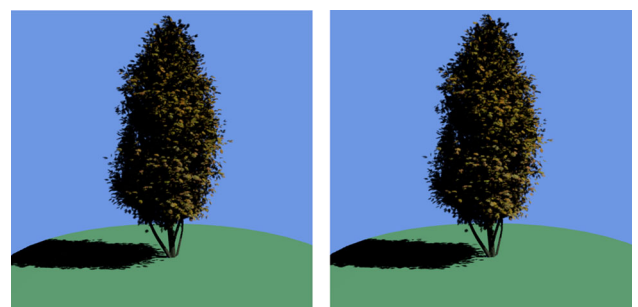


Fig. 7 Cottonwood tree with shadows. *Left* By ray tracing from each shaded point, and *right* precomputed shadows

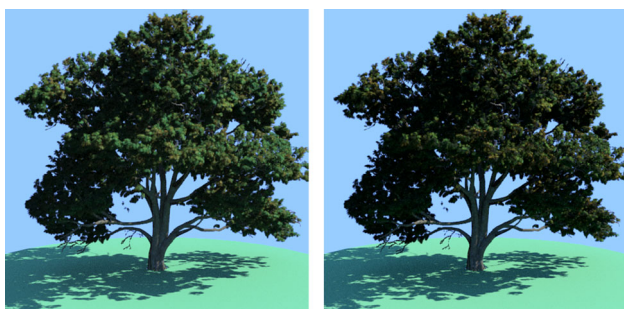


Fig. 8 Path traced beech tree. *Left* reflection plus transmission, *right* transmission only

rays were propagated, we divided the accumulated flux in each leaf cell by the accumulated length and averaged this illumination up the octree hierarchy. This illumination pre-computation took 13.6 s, and then the rendering took 60.1 s. The flux propagation was done in parallel, by dividing the grid of rays up into square subregions indexed by column i and row j , and doing four passes, with pass k including only the squares with $(i \bmod 2) + 2(j \bmod 2) = k$. Squares were assigned to different threads, and since the squares in each pass were well separated, there was no hazard from two threads attempting to simultaneously add to the flux or length in the same octree cell.

Figure 1 shows a forest of 218 trees of various species, rendered by path tracing. The next direction for each path was chosen by sampling a surface normal from the spherical harmonics microflake distribution using only the 15 even terms up to $L = 4$, reversing the normal if it was not facing the viewer, and then sampling from the diffuse BRDF with that normal. Please see the supplemental material videos showing path traced 360 frame cycles with the camera rotating around the center of this forest, using the various methods discussed in this paper, and also a camera dolly showing smooth transition between octree levels.

Figure 8 shows a path traced 500 by 500 image of a beech tree with double-sided microflakes from an SH representation with $L = 4$. The left-hand image includes both the diffuse reflection and the diffuse transmission, while the right-hand image includes only the diffuse transmission, to show its contribution. The main visible effect of the path tracing is the illumination from the sky, particularly evident in the shadows. The supplementary material also includes a path traced video with the camera rotating around this tree.

Figure 9 shows images of a head model with 60,000 hairs, each represented by a polygonal quad mesh on a curved cylinder following a Bezier curve, with 25 segments along the curve and 12 around the circular cross section. They are rendered at 250 by 250 resolution, with 6 by 6 supersamples except for the left-hand ground truth image, in which the original polygonal representation was ray traced at 20 by

20 supersamples. The polygon mesh file was 1,355,638,498 bytes. The images in the top row were rendered from uncompressed octrees. Table 4 gives the octree sizes for these images, and the RMS errors to ground truth and from compression. The construction and rendering times shown in Table 4 were on a 4 core machine. The construction times include precomputing the illumination, and compression, and the compressed rendering times in the last column include decompression. The octree resolution was only 256^3 so that each octree cell intersected multiple hairs, stressing the both the summarization ability of the different representation methods, and also their compression accuracy. Moon et al. [22] also used spherical harmonics for rendering hair, but used them to store the incoming multiple scattered radiance at each cell, as computed from a Monte Carlo simulation, rather than to store the distribution of microflake normals.

Figure 10 shows a sphere, rendered as a hard surface by adjusting the extinction and inscattering computations, at the first nonempty cell the viewing ray encounters, to represent a completely opaque surface with the probability distribution of normals given by the approximated $h(x, \omega_n)$. The hard surface and k means method agree with the “ground truth” image ray traced from the polygonal model. The spherical harmonics method shows a main highlight that is too broad, and also a spurious highlight ring near the profile, caused by a positive ringing lobe in the SH fit, and also by the effect of “double-sided lighting” from the double-sided microflakes. This spurious highlight is mostly removed by the precomputed shadows. The ellipsoidal microflake method is more effective, but must still broaden the surface normal distribution somewhat, because an almost flat ellipsoid generates floating point exceptions or numerical problems in the calculations and cannot be used. Thus, the shadow terminator is softer than in the ground truth image.

9 Discussion

Bruneton and Neyret [1] survey methods of prefiltering surface properties, specifically discussing distributions of microfacet normals in their Sect. 3, and a more general classification of prefiltering methods into three classes in their Sect. 7. In terms of that classification, our method (a) using the variance is a spanning set method, our method (b) is a basis function method, and our method (c) appears to be a moment method, like those of Olano et al. [24] and [25]. However, the ellipsoids of [8] represent projected microflake area, instead of the unprojected area-weighted normals in [25]. As mentioned above, this means that the coefficients do not add linearly when the distributions of child cells are combined into their parent cell. On the other hand, the 3D covariance method of [25] gives the same information as our spherical harmonic method with $L = 2$. In general, since the

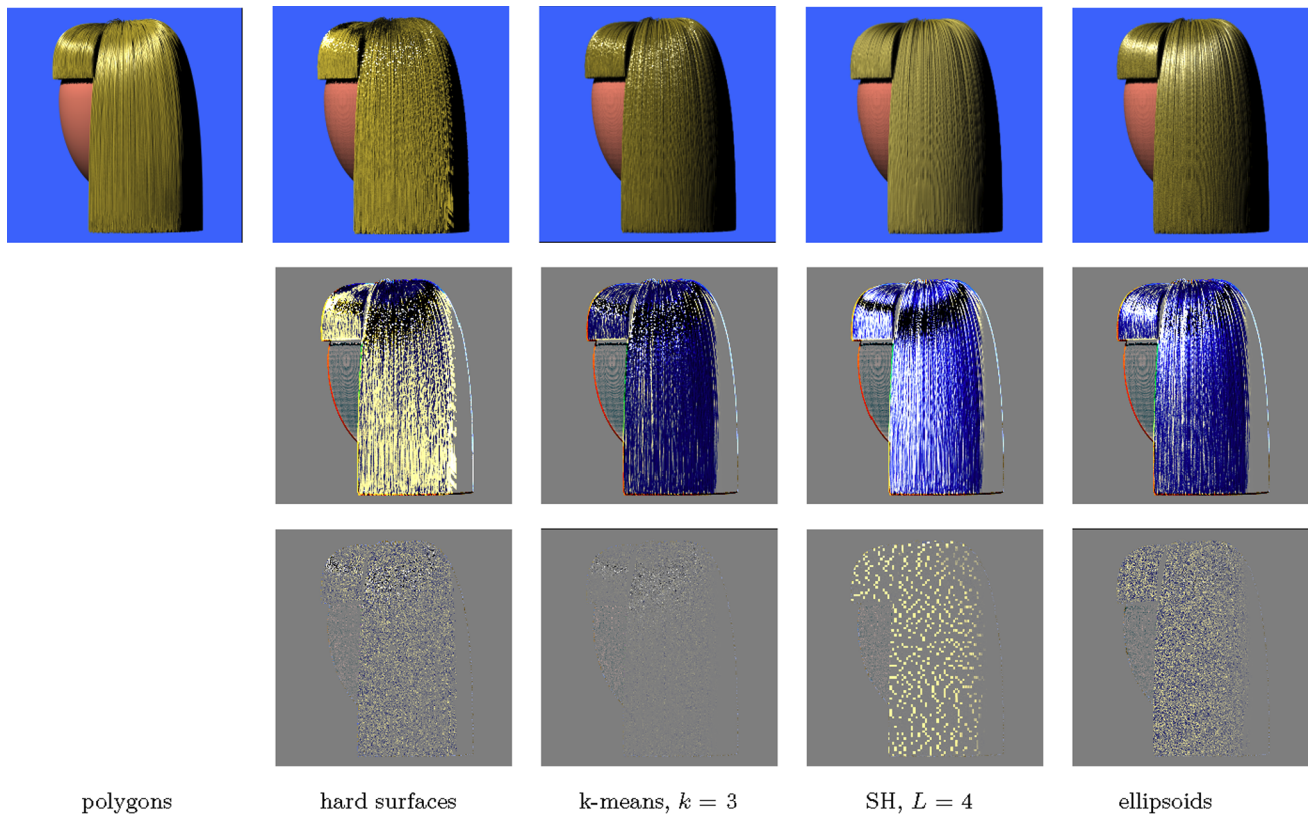


Fig. 9 Head with hair: *top row from left to right* ground truth image ray traced from the polygonal model, hard surfaces, k means with $k = 3$, SH with $L = 4$, and ellipsoidal microflakes. *Middle row* differences between the image in the *top row* and the ground truth image on the left, scaled by a factor of 5, with zero as the neutral gray in the background. *Bottom row* differences between the image rendered from a compressed octree and the corresponding image in the *top row*, scaled by 5

Table 4 Statistics for compression

Method	SVO size	Compr. size	Polygons RMS	Compr. RMS	Constr. tm (s)	Render tm (s)	Compr. rend (s)
Hard surface	165,151,806	64,001,340	0.1293	0.0196	457	3.02	4.76
k means	410,912,420	108,682,498	0.1126	0.0061	596	9.63	86.3
SH: 25 terms	432,539,169	166,440,279	0.0927	0.0467	568	21.1	147.9
Ellipsoidal	388,167,513	84,975,047	0.0976	0.0185	1088	21.1	36.1

Construction time includes SVO construction, precomputed illumination, and compression

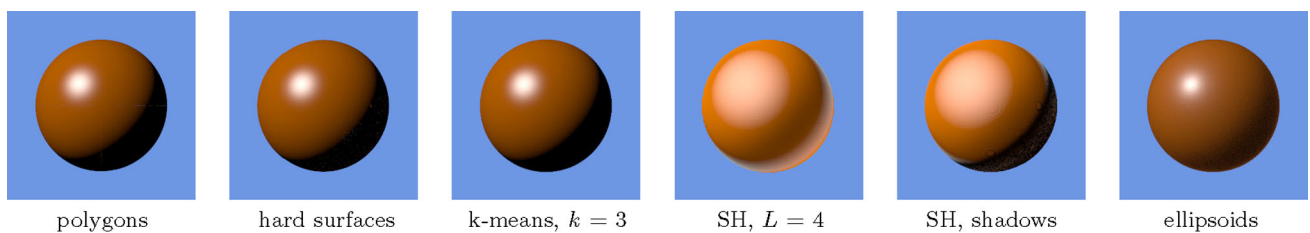


Fig. 10 From left to right a sphere rendered from polygons, hard surfaces, k means with $k = 3$, spherical harmonics with $L = 4$ and only even-ordered terms (showing the effects of double-sided microflakes), spherical harmonics with $L = 4$ and only even-ordered terms with pre-computed shadows to suppress the spurious highlights, and ellipsoidal microflakes

spherical harmonics of order up to L are a basis for the polynomials in x , y , and z on the unit sphere of degree up to L , our method (b) is equivalent to the moment method of [1].

For the cottonwood tree at one sample per pixel and 200 by 200 resolution, simulating a distant view, the $L = 4$ spherical harmonics method has lowest error, although it requires most processing time. However, this test puts the ellipsoid method at a disadvantage, since a single sampled normal per pixel, required because there is no analytic integral for the diffuse reflectance from this distribution, introduces a lot of noise. For the lodgepole pine tree, we used 2 by 2 supersampling, and the noise in the ellipsoid method is less. For this model, the SH method with all 25 terms up to $L = 4$ had the least RMS error, but its SVO used the most storage.

For the top left image in Fig. 6, the optimized Embree ray tracer, running in parallel on the same 12 core machine, took 12.42 s to create its acceleration structure and ray trace the 200 by 400 image at 40 by 40 supersamples per pixel. Thus, the times quoted in Table 3 are not a significant speed improvement. However, the acceleration structure for ray tracing the scene in Fig. 1 with Embree could not fit in memory. The model for Fig. 1 was in fact created from rotated and translated copies of the same four trees, but if every tree was different, its data size would be unmanageable, while even the uncompressed SVO sizes are practical for rendering, varying from 1,670,384,307 bytes for the hard surfaces to 2,717,912,229 bytes for the 25 SH terms up to $L = 4$, at 1024^3 voxel resolution.

The bottom row of difference images in Fig. 9 and the RMS errors from Table 4 show that the compression does not introduce much error. The top row of images shows visible artifacts from the SVO methods, particularly in the highlights. When many hairs are in an octree cell, the single average normal used in the hard surface method cannot adequately represent them. The SH normals are better for diffuse shading, but inadequate to represent the highlights, because of the band-limiting effect of a practical maximum order L . The k means method gives discontinuous shading unless the starting angular phase for the 12 facets around the cylinder is chosen randomly per hair, as was done for Fig. 9, so that there are no preferred representative directions, but then the highlights are noisy. As expected, since the distribution of normals of a long thin ellipsoid is a good approximation to that of coherently oriented hairs, the ellipsoidal method performs best in this case.

The band-limiting effect of the SH approximation makes the highlight too large in Fig. 10 as well as in Fig. 9. The SH method performs better for wider, more general distributions, where concentrated highlights do not occur. The k means method would require more representatives to handle general distributions. When the variance is taken into account, it is similar to a gaussian mixture model. The iterative Expectation Maximization method of fitting a gaussian mixture

Table 5 The suitability of the three approximation methods for different sorts of distributions of microflake normals: + means suitable, – means unsuitable, a blank means suitable, but not the best, and e means could work with enough terms

Distribution type	k means	SH	Ellipsoids
Smooth surface	+	–	
One narrow peak	+	–	+
One wide peak		+	+
Hair cylinders	e		+
Many peaks	e	e	–
General	e	e	–

model might give better approximations for the same data size, but it would take much longer than the k means method. Table 5 summarizes the suitability of the three methods for different sorts of applications.

Publications [9] and [10] consider microfacet height field surface models where the color is correlated with the height and can account for the masking effects related to this correlation during rendering. None of the volume microflake models we consider can account for this sort of volume masking, because we assume that the summarized microflake properties are homogeneous within each octree cell.

Acknowledgements We thank Mark Meyer, Tony DeRose, Eric Heitz, Wojciech Jarosz, Derek Nowrouzezahrai and Ted Kim for helpful discussions, and the SIGGRAPH, Pacific Graphics, and Visual Computer reviewers for useful suggestions. Nelson Max thanks the University of California, Davis for sabbatical salary.

Appendix

In this appendix, we give some formulas for the spherical harmonics calculations discussed above. Good references for the mathematics of spherical harmonics as applied to computer graphics are [28] and [5]. Sloan [28] gives simple formulas for the $Y_{lm}(\omega)$ in the form of polynomials of degree l in the x , y , and z coordinates of the unit vector ω , which we used in our implementation.

To compute the extinction integral in Eq. (3) using an $h(\omega)$ of the form of Eq. (1), we expand the nonnegatively clamped cosine of θ , $A(\theta)$, into spherical harmonics. Since this function is independent of ϕ , all the coefficients with $m \neq 0$ are zero so we get just the “zonal” harmonics expansion

$$A(\theta, \phi) \approx \sum_{l=0}^L A_{l0} Y_{l0}(\theta, \phi) \quad (12)$$

where, by Eq. (2),

$$A_{l0} = 2\pi \int_0^{\pi/2} Y_{l0}(\theta, \phi) \cos\theta \sin\theta \, d\theta \, d\phi. \quad (13)$$

Ramamoorthi and Hanrahan [26] (please see the correction of their Eq. (19) at the bottom of the web page <http://cseweb.ucsd.edu/ravir/papers/invlamb/>) give the formula for the coefficients A_{l0} , which are zero for odd $l > 1$, and decrease rapidly for increasing even l . We have used only the first few values $A_{00} = \sqrt{\pi}/2$, $A_{10} = \sqrt{\pi}/3$, $A_{20} = \sqrt{5\pi}/8$, $A_{30} = 0$, and $A_{40} = -\sqrt{\pi}/16$.

This zonal harmonic expansion is only useful for evaluating Eq. (3) if ω_o is at the north pole of the unit sphere. For other directions of ω_o , one must rotate this zonal harmonic expansion so that its axis of symmetry is along the direction ω_o . This is simpler for zonal harmonics than for general spherical harmonics. According to [26], the result of rotating the expansion in equation (12) to make the axis of symmetry lie in the direction $\omega_o = (\theta_o, \phi_o)$ is

$$R^{\theta_o\phi_o} A(\theta, \phi) \approx \sum_{l=0}^L \sum_{m=-l}^l Y_{lm}(\theta_o, \phi_o) \sqrt{\frac{4\pi}{2l+1}} A_{l0} Y_{lm}(\theta, \phi). \tag{14}$$

Thus, if Eq. (1) is the spherical harmonic expansion of the microflake normal density $h(\omega)$, by equation (3) and the orthonormality of the SH basis functions,

$$\sigma_i(x, \theta_o, \phi_o) \approx \sum_{l=0}^L \sum_{m=-l}^l Y_{lm}(\theta_o, \phi_o) \sqrt{\frac{4\pi}{2l+1}} A_{l0} h_{lm}. \tag{15}$$

The inscattering integral in Eq. (4) contains the clamped cosine $\langle \omega_n, \omega_i \rangle$ as well as $\langle \omega_n, \omega_o \rangle$, so if $\langle \omega_n, \omega_i \rangle$ is also expanded in spherical harmonics as in Eq. (14), we get, for the perfectly diffuse reflection of a small light source from direction ω_i of radiance times solid angle equal to I ,

$$\begin{aligned} S(\omega_o) &= \int_{\Omega} \int_{\Omega} h(\omega_n) \frac{\alpha}{\pi} \langle \omega_n, \omega_i \rangle \langle \omega_n, \omega_o \rangle L(\omega_i) d\omega_n d\omega_i \\ &= \frac{I\alpha}{\pi} \int_0^{2\pi} \int_0^\pi h(\theta, \phi) R^{\theta_i\phi_i} A(\theta, \phi) R^{\theta_o\phi_o} A(\theta, \phi) \sin\theta d\theta d\phi \\ &\approx \frac{I\alpha}{\pi} \int_0^{2\pi} \int_0^\pi \sum_{l=0}^L \sum_{m=-l}^l h_{lm} Y_{lm}(\theta, \phi) \\ &\quad \sum_{l'=0}^L \sum_{m'=-l'}^{l'} Y_{l'm'}(\theta_o, \phi_o) \sqrt{\frac{4\pi}{2l'+1}} A_{l'0} Y_{l'm'}(\theta, \phi) \sum_{l''=0}^L \\ &\quad \sum_{m''=-l''}^{l''} Y_{l''m''}(\theta_i, \phi_i) \sqrt{\frac{4\pi}{2l''+1}} A_{l''0} Y_{l''m''}(\theta, \phi) \sin\theta d\theta d\phi \end{aligned}$$

$$\begin{aligned} &= \frac{I\alpha}{\pi} \sum_{l=0}^L \sum_{m=-l}^l h_{lm} \sum_{l'=0}^L \sum_{m'=-l'}^{l'} Y_{l'm'}(\theta_o, \phi_o) \sqrt{\frac{4\pi}{2l'+1}} \\ &A_{l'0} \sum_{l''=0}^L \sum_{m''=-l''}^{l''} Y_{l''m''}(\theta_i, \phi_i) \sqrt{\frac{4\pi}{2l''+1}} A_{l''0} T_{lm'l''m''} \end{aligned} \tag{16}$$

where $T_{lm'l''m''}$ is the ‘‘triple product integral’’

$$T_{lm'l''m''} = \int_0^{2\pi} \int_0^\pi Y_{lm}(\theta, \phi) Y_{l'm'}(\theta, \phi) Y_{l''m''}(\theta, \phi) \sin\theta d\theta d\phi. \tag{17}$$

For order $L = 4$, the potential number of triple product integrals is $25^3 = 15,625$, not accounting for symmetries in the indices, but only 1,158 of them are nonzero, and these are precomputed and stored in a sparse table. In fact, since $A_{30} = 0$, only 605 terms are actually included in the sum in Eq. (16), and if the microflakes are double-sided, and we approximate $|\cos(\theta)|$ by $A(\theta, \phi) + A(-\theta, \phi)$, all terms with $l' = 1$ or $l'' = 1$ also disappear, and only 585 terms remain.

Note that if $\omega_o = (\theta_o, \phi_o)$ is fixed, grouping the terms and factors from $h(\omega_n) \langle \omega_n, \omega_o \rangle$ in the last form in Eq. (16) into the large parentheses below gives a spherical harmonic expansion in the direction variable $\omega_i = (\theta_i, \phi_i)$.

$$\begin{aligned} S(\omega_o) &= \sum_{l''=0}^L \sum_{m''=-l''}^{l''} \left(\frac{I\alpha}{\pi} \sum_{l=0}^L \sum_{m=-l}^l h_{lm} \sum_{l'=0}^L \sum_{m'=-l'}^{l'} Y_{l'm'}(\theta_o, \phi_o) \right. \\ &\quad \left. \times \sqrt{\frac{4\pi}{2l'+1}} A_{l'0} \sqrt{\frac{4\pi}{2l''+1}} A_{l''0} T_{lm'l''m''} \right) Y_{l''m''}(\theta_i, \phi_i) \end{aligned} \tag{18}$$

Thus, it can be importance-sampled for path tracing using the techniques of [14], or by rejection sampling. Note also from this form that once the basis function coefficients in the large parentheses have been computed, the inscattering of illumination from multiple light sources can be computed with only $(L + 1)^2$ multiplications and adds for each. For a light source ‘‘at infinity’’ with constant ω_i , a different grouping of the terms involving instead the factors $\langle \omega_n, \omega_i \rangle \langle \omega_n, \omega_o \rangle$ can be computed once per viewing ray that hits microflakes, allowing only $(L + 1)^2$ multiplications per octree cell (9.2s for the image in Fig. 9 instead of 21.1). For finite distance light sources, using more storage, the inscattering and the extinction coefficient can be computed once per voxel as first needed and saved for subsequent viewing rays crossing the same voxel (8.4s for the image in Fig. 9).

References

1. Bruneton, E., Neyret, F.: A survey of non-linear pre-filtering methods for efficient and accurate surface shading. *IEEE Trans. Vis. Comput. Gr.* **18**(3), 242–260 (2008)
2. Cook, R., Carpenter, L., Catmull, E.: The Reyes image rendering architecture. In: *SIGGRAPH '87: proceedings of the 14th annual conference on computer graphics and interactive techniques*, pp. 95–102. ACM (1987). doi:[10.1145/37402.37414](https://doi.org/10.1145/37402.37414)
3. Cook, R., Torrance, K.: A reflectance model for computer graphics. *ACM Trans. Gr.* **1**(1), 7–12 (1982). doi:[10.1145/357290.357293](https://doi.org/10.1145/357290.357293)
4. Crow, F.: A more flexible image generation environment. In: *ACM SIGGRAPH '82: Proceedings of the 9th annual conference on computer graphics and interactive techniques*, vol. 16, pp. 9–18 (1982). doi:[10.1145/800064.801253](https://doi.org/10.1145/800064.801253)
5. Green, R.: Spherical harmonic lighting: the gritty details. In: *Archives of the Game Developers Conference*, vol. 56 (2003)
6. Heitz, E.: Understanding the masking-shadowing function in microfacet-based brdfs. *J. Comput. Gr. Tech.* **3**(2), 32–91 (2014)
7. Heitz, E., Dupuy, J., Crassin, C., Dachsbacher, C.: The SGGX microflake distribution: supplementary material. ACM Digital Library (2015)
8. Heitz, E., Dupuy, J., Crassin, C., Dachsbacher, C.: The SGGX microflake distribution. *ACM Trans. Gr.* **34**(4), 48 (2015)
9. Heitz, E., Neyret, F.: Representing appearance and pre-filtering subpixel data in sparse voxel octrees. In: Dachsbacher, C., Munkberg, J., Pantaleoni, J. (eds.) *Eurographics/ ACM SIGGRAPH symposium on high performance graphics. The Eurographics Association* (2012). doi:[10.2312/EGGH/HPG12/125-134](https://doi.org/10.2312/EGGH/HPG12/125-134)
10. Heitz, E., Nowrouzezahrai, D., Poulin, P., Neyret, F.: Filtering color mapped textures and surfaces. In: *Proceedings of I3D*, pp. 129–136. ACM (2013)
11. Hoppe, H.: Progressive meshes. In: *ACM SIGGRAPH '96: proceedings of the 24th annual conference on computer graphics and interactive techniques*, pp. 99–108. ACM Press (1996). doi:[10.1145/237170.237216](https://doi.org/10.1145/237170.237216)
12. Hoppe, H.: View-dependent refinement of progressive meshes. In: *ACM SIGGRAPH '97: proceedings of the 24th annual conference on computer graphics and interactive techniques*, pp. 189–198. ACM Press/Addison-Wesley Publishing Co., New York (1997). doi:[10.1145/258734.258843](https://doi.org/10.1145/258734.258843)
13. Jakob, W., Arbee, A., Moon, J.T., Bala, K., Marschner, S.: A radiative transfer framework for rendering materials with anisotropic structure. *ACM Transactions on Graphics (TOG)—proceedings of ACM SIGGRAPH 2010* **29**(4) (2010). doi:[10.1145/1778765.1778790](https://doi.org/10.1145/1778765.1778790)
14. Jarosz, W., Carr, N., Jensen, H.W.: Importance sampling spherical harmonics. *Comput. Gr. Forum* **28**(2), 577–586 (2009). doi:[10.1111/j.1467-8659.2009.01398.x](https://doi.org/10.1111/j.1467-8659.2009.01398.x)
15. Jensen, H.W.: *Realistic Image Synthesis Using Photon Mapping*. A K Peters/CRC Press, Boca Raton (2012)
16. Joy, K.I., Grant, C.W., Max, N.L., Hatfield, L.: Tutorial: computer graphics: image synthesis. ACM Press, p 2 (1988)
17. Kainz, F., Bogart, R.: <https://github.com/openexr/openexr/tree/master/ilmbase/half> (2002)
18. Kanungo, T., Mount, D.M., Netanyahu, N., Piatko, C., Silverman, R., Wu, A.Y.: A local search approximation algorithm for k-means clustering. *Comput. Geom. Theory Appl.* **28**, 89–112 (2004)
19. Kaplanyan: Cryengine 3: reaching the speed of light. In: *SIGGRAPH 2010: advances in Real-time Rendering course* (2010)
20. Max, N.: Optical models for direct volume rendering. *IEEE Trans. Vis. Comput. Gr.* **1**, 99–108 (1995)
21. Max, N., Mobley, C., Wu, E.H.: Plane-parallel radiance transport for global illumination in vegetation. In: *Rendering techniques '97: proceedings of eurographics workshop on rendering*, pp. 239–250. Springer, Berlin (1997)
22. Moon, J., Walter, B., Marschner, S.: Efficient multiple scattering in hair using spherical harmonics. *ACM Trans. Gr.* **27**(3), 1–7 (2008)
23. Neyret, F.: Modeling animating and rendering complex scenes using volumetric textures. *IEEE Trans. Vis. Comput. Gr.* **4**(1), 55–70 (1998). doi:[10.1109/2945.675652](https://doi.org/10.1109/2945.675652)
24. Olano, M., Baker, D.: Lean mapping. In: *Proceedings of I3D'10*, pp. 181–188. ACM (2010)
25. Olano, M., North, M.: Normal distribution mapping. Tech. Rep. TR97-041, Department of Computer Science, University of North Carolina, Chapel Hill (1997)
26. Ramamoorthi, R., Hanrahan, P.: On the relationship between radiance and irradiance: determining the illumination from images of a convex Lambertian object. *J. Opt. Soc. Am. A* **18**(10), 2448–2459 (2001)
27. Rense, W.: Polarization studies of light diffusely reflected from ground and etched glass surfaces. *J. Opt. Soc. Am.* **40**(1), 55–59 (1950). doi:[10.1364/JOSA.40.000055](https://doi.org/10.1364/JOSA.40.000055)
28. Sloan, P.P.: Stupid spherical harmonics (sh) tricks. Microsoft Corporation (2008). Expanded material from presentation at the 2008 Game Developer Conference. <http://www.ppsloan.org/publications/StupidSH36.pdf>
29. Toksvig, M.: Mipmapping normal maps. *J. Gr. Tools* **10**(3), 65–71 (2005)
30. Torrance, K., Sparrow, E.: Theory for off-specular reflection from roughened surfaces. *J. Opt. Soc. Am.* **57**(9), 1105–1114 (1967)
31. Walter, B., Marschner, S.R., Li, H., Torrance, K.E.: Microfacet models for refraction through rough surfaces. In: Kautz, J., Pattanaik, S. (eds.) *Rendering techniques*, pp. 105–205. The Eurographics Association (2007). doi:[10.2312/EGWR/EGSR07/195-206](https://doi.org/10.2312/EGWR/EGSR07/195-206)
32. Zhao, S.: Modeling and rendering fabrics at micron resolution. Ph.D. thesis, Cornell University (2014)
33. Zhao, S., Jacob, W., Marchner, S., Bala, K.: Building volumetric appearance of fabrics using micro CT imaging. *ACM Trans. Gr.* **30**(4), 44:1–44:10 (2011)



Nelson Max received his Ph.D. in Mathematics from Harvard University in 1967 and is currently a Distinguished Professor of Computer Science at the University of California, Davis. His research interests are in the areas of scientific visualization, computer animation, realistic computer graphics rendering, and multi-view stereo reconstruction. In visualization, he works on molecular graphics, and volume and flow visualization, particularly on irregular finite element meshes. He has rendered realistic lighting effects in clouds, trees, and water waves and has produced numerous computer animations, shown at the annual ACM SIGGRAPH conferences, and in OMNIMAX stereo at the Fujitsu Pavilions at Expo '85 in Tsukuba Japan, and Expo '90 in Osaka Japan. He is a fellow of the Association for Computing Machinery, and the winner of its 2007 Steven A. Coons award.



Tom Duff was educated at the universities of Waterloo and Toronto and has worked at the NYIT Computer Graphics Lab, the Lucasfilm computer division, Bell Labs and, currently, Pixar. His main research interest is in computer graphics and animation, but he has also published in operating systems, polymer physics, wireless networking, mathematics and computer security. Past projects include writing (with Bill Reeves) the first version of Pixar's animation system,

special effects in *Star Trek II*, *Return of the Jedi* and *A Bug's Life*, a full-scale radio-controlled Walle and a light-field movie camera. He has a dozen patents and two Academy Awards® for contributions to motion picture science and technology. When not at work, he is an opera singer, musical tinkerer and grandparent.



Ben Mildenhall is a Ph.D. student at the University of California, Berkeley, where his work is supported by a Fannie and John Hertz Foundation fellowship. His current research interests are 3D reconstruction for virtual reality, structured light capture, and light field reconstruction. He received his undergraduate degree from Stanford University in 2015, where his work focused on applications of probabilistic programming to content generation.



Yajie Yan is a Ph.D. student at Washington University in St Louis. His research at school focuses on geometry processing, specifically robust computation and applications of medial axis. He obtained his master degree from Beihang University in 2012, working on feature extraction from 3D point clouds. Besides geometry, he is also interested in realistic rendering and 3D fusion applied in VR/AR contexts.