A practical stochastic algorithm for rendering mirror-like flakes

Conference Paper · July 2016

DOI: 10.1145/2897839.2927391

2 authors:

Asen Atanasov
Chaos Group, Bulgaria, Sofia
8 PUBLICATIONS **8** CITATIONS

SEE PROFILE

Vladimir N. Koylazov
Chaos Software Ltd
15 PUBLICATIONS **16** CITATIONS

SEE PROFILE

A Practical Stochastic Algorithm for Rendering Mirror-Like Flakes

Asen Atanasov   Vladimir Koylazov
Chaos Group*

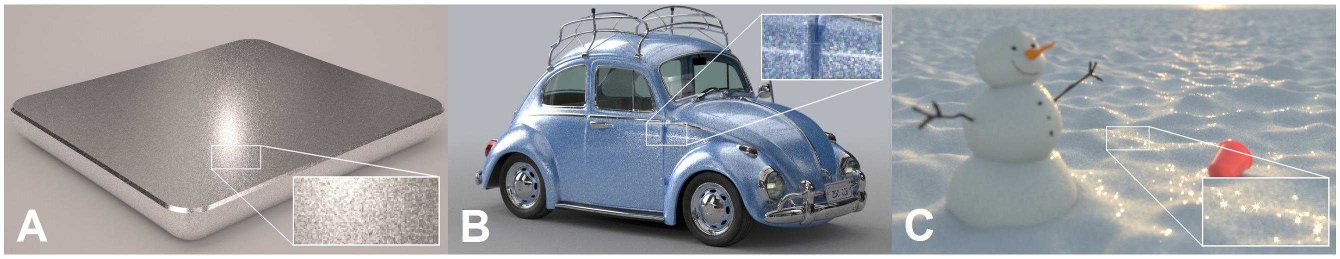

Figure 1: *A) A metal plate with our flakes material. B) A Beetle toy with diffuse base layer, stochastic flakes middle layer, and reflective coat. Flakes use triplanar mapping and have colors with random hue in HSL color space. C) Sparkling snow with sub-surface scattering base layer and stochastic flakes coat material. The model parameters* $(N, \alpha_{GGX}, \gamma)$ *for A), B), and C) are* $(16 \times 10^7, 0.04, 1°)$, $(8 \times 10^7, 0.09, 0.3°)$, *and* $(10^8, 0.25, 0.4°)$ *respectively. B) and C) have ray-traced depth of field and slight post-process lens effects.*

Abstract

Materials, such as snow, sand, metallic paints, rough plastics, and metals, often exhibit small-scale phenomena observed as bright sparkling or glittering surface features. These features become more pronounced under narrow-angle illumination and vary based on the orientation of the surface with respect to the viewer and light sources. Microfacet-based surface models, composed of a large finite number of microscopic mirror-like flakes, can mimic this effect. An associated microfacet BRDF and a memory-efficient stochastic algorithm are explored in [Jakob et al. 2014]. We present a new stochastic algorithm that inherits the good properties of the original algorithm, but does not require any precomputation; implements optimal importance sampling which is extended to efficiently sample wide and heavy-tailed microfacet distributions (i.e. GGX), and offers better overall performance. In addition, a triplanar mapping technique is employed to handle geometry without texture coordinates. The algorithm is both practical and easier to implement.

Keywords: Stochastic algorithm, microfacet BRDF, importance sampling, sparkling snow, metallic paint

Concepts: •**Computing methodologies** → **Rendering; Reflectance modeling;**

1 Introduction

Microfacet theory idealizes surfaces as composed of infinitely many microscopic facets, described by two statistical measures - the microfacet distribution $D(\mathbf{x}, \mathbf{h})$ and the shadowing-masking function $G(\mathbf{i}, \mathbf{o}, \mathbf{h})$. The related microfacet BRDF is $f_r(\mathbf{x}, \mathbf{i}, \mathbf{o}) = \frac{1}{4(\mathbf{i}\cdot\mathbf{n})(\mathbf{o}\cdot\mathbf{n})} F(\mathbf{i} \cdot \mathbf{h}) D(\mathbf{x}, \mathbf{h}) G(\mathbf{i}, \mathbf{o}, \mathbf{h})$, where F is the Fresnel term, $\mathbf{x}$ - the shading point, $\mathbf{i}$ - direction from which light is incident, $\mathbf{o}$ - direction in which light is reflected, $\mathbf{h} = \frac{\mathbf{i}+\mathbf{o}}{||\mathbf{i}+\mathbf{o}||}$, and $\mathbf{n}$ - the surface normal [Walter et al. 2007]. Converged renders of such surfaces appear perfectly smooth. However, the visible microstructure, especially observed in motion or change of illumination, gives more realism. Usually, the rendering of small specular surface features is achieved through normals maps, but as the features become much smaller than a pixel, resolving the correct pixel value using Monte Carlo sampling becomes inefficient.

2 Previous work

The memory-efficient rendering of large finite number of flakes is investigated in [Jakob et al. 2014]. N flakes are uniformly distributed in the unit texture space and their normals follow a microfacet distribution on the unit hemisphere. The multiscale BRDF is defined as the microfacet BRDF, averaged over a finite surface area A, containing the shading point $\mathbf{x}$ and a finite solid angle Ω around the incident direction $\mathbf{i}$. In the implementation, A is a parallelogram approximation of the pixel footprint, defined by the ray differentials [Idehy 1999], and Ω is a cone of radius γ, centered at $\mathbf{i}$. The continuous microfacet distribution D is substituted by a discrete counterpart $\frac{4}{a(A)\sigma(\Omega)}(\mathbf{i} \cdot \mathbf{h})\widehat{D}$, where $\widehat{D}$ is the fraction of flakes which are contained in A and reflect $\mathbf{o}$ in Ω. The expression $(\mathbf{i} \cdot \mathbf{h})\widehat{D}$ is an approximation to $\tilde{D} = \frac{1}{N}\sum \mathbf{i} \cdot \mathbf{m}_j$, where $\mathbf{m}_j$ are the normals of this fraction of flakes. A hierarchical traversal of the texture-direction space, starting from the unit texture square and the unit hemisphere, is performed to compute $\widehat{D}$. A key advantage of the algorithm is that the flakes are not stored in memory, but their counts are reproduced by a deterministically seeded stochastic process during traversal. The result converges to a standard microfacet BRDF as N increases. This method has practical issues. To build the data structure efficiently on average, it relies on statistics collected during a full animation precomputation step. The importance sampling strategy does not sample the discrete integrand, but samples from a different smooth distribution. Thus, it practically fails to sample wide and heavy-tailed microfacet distributions.

3 Our approach

We present a new stochastic algorithm, based on [Jakob et al. 2014], which inherits the good properties and solves the issues listed above. For a given shading point with a footprint A, we per-

*e-mail: {asen.atanasov,vlado}@chaosgroup.com

SIGGRAPH '16, July 24-28, 2016, Anaheim, CA,
ISBN: 978-1-4503-4282-7/16/07
DOI: http://dx.doi.org/10.1145/2897839.2927391

form one texture space query to locate the number of flakes in it during which we explicitly generate the normals of the flakes and collect data to calculate the lighting. A comparison between the two algorithms is given in Table 1.

Texture query. We query the texture space with the footprint parallelogram A, using a quad-tree, stack-based, depth-first traversal. The root of the tree is the unit texture square which has the flakes count N and seed s used to initialize a pseudorandom generator for this query. Each square node is split into 4 equal nodes. Node-footprint intersections are performed to filter out the nodes which do not overlap with the footprint. On each split operation we distribute the node flakes count to its children using normal approximation to multinomial distribution with probabilities $\mathbf{p} = (\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4})$ [Jakob et al. 2014]. Our query generator combines four xorshift-based generators which share a 64-bit state. During the traversal push operation, we use a different generator for each child node and the state is stored on the stack. On the pop operation, the generator state is restored. A node is a leaf if it has less than K flakes, or the stack depth T is reached. For our implementation $N < 2^{31}$ and $(K, T) = (16, 15)$ proved adequate. If a leaf is entirely inside the footprint, then all of its flakes are processed. Otherwise we generate random positions of the flakes inside the leaf and process these flakes which belong to the footprint.

Reflection cache. During the query, we accumulate n footprint flakes and generate their random normals $\mathbf{m}_j$, according to the desired microfacet distribution, using the query generator. Then $\mathbf{o}$ is reflected from each normal $\mathbf{m}_j$ to get per-flake directions $\mathbf{r}_j$. These directions represent n cones of radius γ from which light can be reflected along the outgoing direction $\mathbf{o}$. Furthermore, we define per-flake weights $w_j = \mathbf{m}_j \cdot \mathbf{r}_j$. We call the set of all n pairs $(\mathbf{r}_j, w_j)$ reflection cache. For the microfacet distribution, we compute an accurate approximation $\tilde{D} \approx \frac{1}{N} \sum_{\mathbf{i}\cdot\mathbf{r}_j \geq \cos\gamma} w_j$ by checking how many cached cones contain the incident light direction $\mathbf{i}$.

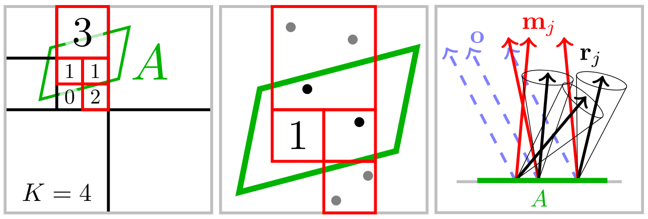

Figure 2: *Four non-empty leaves (red) overlapping the footprint (left). Partially overlapping leaves generate flakes positions and check if they belong to the footprint (middle). Generated normals* $\mathbf{m}_j$ *for the three flakes and reflection cache directions* $\mathbf{r}_j$ *(right).*

Optimal importance sampling. Along with the reflection cache, we construct a cumulative table with $n + 1$ elements $c_0 = 0$, $c_j = c_{j-1} + w_j$, $j = \overline{1, n}$. We pick a random cache direction $\mathbf{r}$, proportional to w_j, by generating a uniform random number $\xi \in (0, c_n)$ and binary searching the cumulative table with it. Then we generate the BRDF direction $\mathbf{i}$ as a uniform random direction in a cone of radius γ around $\mathbf{r}$. The corresponding geometric probability is $\frac{1}{\pi(1-\cos\gamma)}$, where $\pi(1 - \cos\gamma)$ is the solid angle of the cone. Finally, the probability of generating $\mathbf{i}$ is $p(\mathbf{i}) = \frac{k}{\pi(1-\cos\gamma)c_n}$, $k = \sum_{\mathbf{i}\cdot\mathbf{r}_j \geq \cos\gamma} w_j$ is the sum of the weights of the cached cones which contain $\mathbf{i}$ and could have generated it proportionally to their weights. We compute k as a by-product of the BRDF evaluation.

Distance blending. As the number of flakes within the footprint increases, the material appearance conforms to the smooth microfacet model and the calculations of our algorithm increase. Therefore, we implement blending based on the expected number of flakes per pixel footprint $E[n] = N \cdot a(A)$. We define blending range $[b_{min}, b_{max}]$. For all footprints with expected number of flakes $E[n] \leq b_{min}$ we compute only the discrete microfacet distribution; for $E[n] \geq b_{max}$ we compute only the smooth microfacet distribution; for the case of $b_{min} \leq E[n] \leq b_{max}$ we compute both and linearly interpolate between them, using $E[n]$ as an interpolation parameter. Our default blending range is $[500, 2000]$.

Table 1: *A comparison between Jakob et al. and our method for* $N = 2 \times 10^7$ *and distribution width* α. *Variance-based image sampler with a fixed threshold is used to render* 300×200 *region of the metal plate on Intel Core i7-980X, 3.33GHz, 6-core machine.*

α	0.01	0.1	0.4
Jakob et al. (Beckmann, $\gamma = 1°$)	19s	631s	1945s
Jakob et al. (Beckmann, $\gamma = 5°$)	18s	49s	170s
Our method (Beckmann, $\gamma = 1°$)	4s	8s	10s
Our method (Beckmann, $\gamma = 5°$)	6s	8s	10s
Our method (GGX, $\gamma = 1°$)	5s	9s	9s
Our method (GGX, $\gamma = 5°$)	6s	9s	9s

Colored flakes. Optionally, we generate a color table with a fixed number of entries which is either spanning the hue parameter in HSL space for a fixed saturation and lightness, or it is sampled from a texture. Per-flake color is assigned by generating a random index in this table. We use the same blending strategy, but we multiply the smooth BRDF by the average color of the table. For importance sampling we pick random cache direction proportional to the product of the weights of the flakes w_j and their color intensities. In our implementation the table size is 64.

Triplanar mapping. In practice, texture coordinates are not always available or have bad quality. Alternatively, our method works with triplanar mapping. We transform the surface normal and the footprint, defined in world coordinates, into the local coordinates of the shaded object. Then we project the footprint along the axis corresponding to the maximal absolute component of the local normal. Then we scale the resulting 2D footprint depending on the shaded object size. We find the square with integer coordinates and size 1, to which the footprint belongs, and perform the texture query in it. In order to avoid tiling, we generate the seed s of the root node based on the square coordinates. If the footprint is at the boundary of such squares, we query all of them.

Acknowledgements

We thank Stephen Parker, Rossen Dimov, Yordan Madzhunkov, Simeon Balabanov, Kalina Panteleeva, Yavor Stoikov, Emanuele Lecchi, Christopher Nichols and Rhys Dippie.

References

IDEHY, H. 1999. Tracing ray differentials. *In Proceedings of SIGGRAPH 99, ACM Press/Addison-Wesley Publishing Co.*

JAKOB, W., HAAN, M., YAN, L.-Q., LAWRENCE, J., RAMAMOORTHI, R., AND MARSCHNER, S. 2014. Discrete stochastic microfacet models. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2014) 33*, 4.

WALTER, B., MARSCHNER, S., AND TORRANCE, K. 2007. Microfacet models for refraction through rough surfaces. *In Proceedings of the 18th Eurographics Conference on Rendering Techniques*, 195–206.