Pacific Graphics 2020
E. Eisemann, A. Jacobson, and F.-L Zhang
(Guest Editors)

Volume 39 (2020), Number 7

Procedural Physically based BRDF for Real-Time Rendering of Glints

X. Chermain[1], B. Sauvage[1], J.-M. Dischler[1] and C. Dachsbacher[2]

[1]ICube, Université de Strasbourg, CNRS, France
[2]Institute for Visualization and Data Analysis, Karlsruhe Institute of Technology

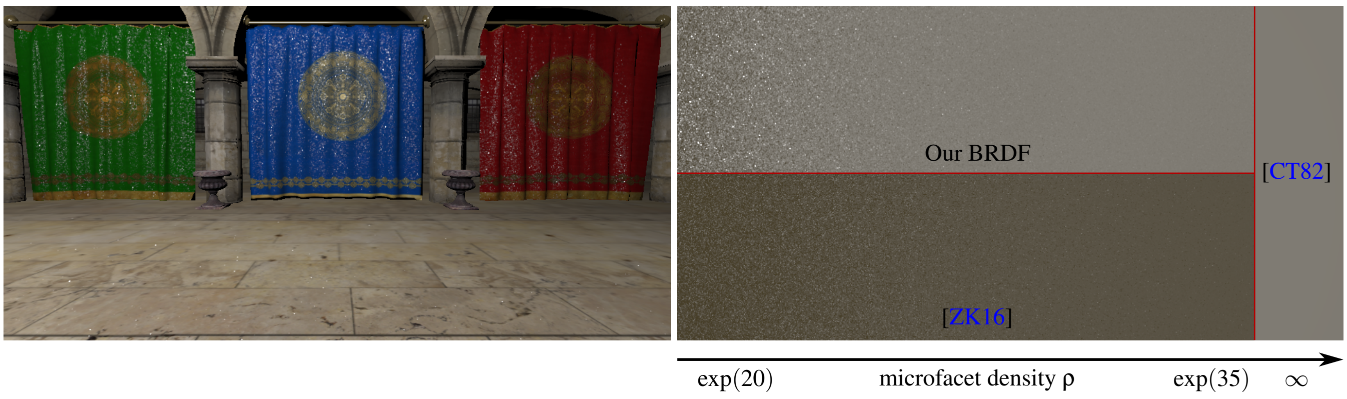

Figure 1: *Left: Sponza scene with sparkling fabrics and stones is rendered on an NVIDIA GeForce RTX 2080 (3.0 ms/frame). Right: A sparkling plane perpendicular to the camera with specular microfacet density increasing from left to right. Top right: our physically based BRDF (2.5 ms/frame) converges to Cook and Torrance's BRDF (0.4 ms/frame), which assumes an infinite number of microfacets. Bottom right: the state-of-the-art method [ZK16] (1.3 ms/frame) is not physically based and does not converge to the BRDF of Cook and Torrance.*

Abstract

Physically based rendering of glittering surfaces is a challenging problem in computer graphics. Several methods have proposed off-line solutions, but none is dedicated to high-performance graphics. In this work, we propose a novel physically based BRDF for real-time rendering of glints. Our model can reproduce the appearance of sparkling materials (rocks, rough plastics, glitter fabrics, etc.). Compared to the previous real-time method [ZK16], which is not physically based, our BRDF uses normalized NDFs and converges to the standard microfacet BRDF [CT82] for a large number of microfacets. Our method procedurally computes NDFs with hundreds of sharp lobes. It relies on a dictionary of 1D marginal distributions: at each location two of them are randomly picked and multiplied (to obtain a NDF), rotated (to increase the variety), and scaled (to control standard deviation/roughness). The dictionary is multiscale, does not depend on roughness, and has a low memory footprint (less than 1 MiB).

CCS Concepts
• ***Computing methodologies*** → ***Reflectance modeling;***

1. Introduction

Photo-realism in real-time is a core topic of recent computer graphics research and has been ever since. However, in spite of drastic improvements of graphics hardware (GPU), it remains highly challenging because of the very strong memory and time constraints it imposes to both global light transport and local shading computation. Local shading reproduces the appearance of the actual material that objects are made of. Its basis is the spatially-varying bi-directional reflectance distribution function (SV-BRDF), a complex 6D function, modelling how individual surface points reflect the light received. Most frequently, its representation is built upon statistics at the sub-pixel level, these statistics modelling the surface non-visible micro-scale geometry and roughness. A common statistical representation is the Normal Distribution Function (NDF) corresponding to sets of microfacets. In the context of photo-realism, the physical validity of the BRDF is mandatory. Concretely, it must respect Helmholtz reciprocity and energy conservation. Often this can be obtained easily for smooth single scale materials, where statistics are defined by single lobe functions assuming an infinite number of microfacets. Conversely, high-frequency specular materials, like glittery materials such as metallic paints, plastics, or sparkling fabrics depict strong angular and spatial variations. They are represented by multi-lobes and multi-scale functions, which are much more challenging because physical accuracy and visual coherence are hard to maintain through all levels of scales. “Explicit” representations, e.g. normal maps, are typically too time and memory consuming for such fine details, whereas computationally efficient procedural representations lack physical validity. Physically based BRDFs save artists time by avoiding post-rendering corrections. Besides, some rendering engines need BRDFs that conserve energy so that their mathematical model is consistent as a whole.

In this paper, we introduce a new, procedural and physically based BRDF to reproduce the appearance of sparkling materials, such as flakes or coarse rough metal and plastic. It is the first glitter material model that unifies efficient computation, extreme compactness and physical validity. The user controls the surface roughness, but also the density of microfacets on the surface. When the number of microfacets increases, our model converges towards a standard smooth BRDF based on a continuous NDF characterized by a single lobe (infinite number of microfacets).

As for previous works, our BRDF uses a $\mathcal{P}$-NDF, which is a NDF defined for a patch $\mathcal{P}$ on the surface. The patch notation $\mathcal{P}$ represents the screen pixel footprint on this surface. Our $\mathcal{P}$-NDF is modelled by a weighted sum of high-frequency NDFs, defined for each cell of a spatially unbounded MIP hierarchy. At level zero, i.e., at the finest level of detail (LOD), NDFs have few lobes and represent few discrete microfacets. Lobes are progressively and consistently introduced as hierarchy level increasing. At the last LOD, the NDF is a standard Beckmann distribution, constructed by aggregating all individual lobes of the lower levels. The novelty of our approach is that our NDFs are generated on-the-fly by multiplying two random marginal distributions, defined in a lower dimension and uniformly picked from a pre-defined dictionary. This permits us to maintain very high performance and extreme compactness. Each of these NDFs is rotated and then scaled. The rotation is random for each MIP hierarchy cell. It increases the variety of NDFs that can be generated using the dictionary and thus avoids rendering artefacts. The scaling allows to use a single generic dictionary independent of roughness: the user can set a target BRDF roughness in real-time, or make it vary spatially. Physical validity is also guaranteed at all levels of scales by an accurate normalisation.

In summary, our contribution is as follows:

- We present a novel physically based BRDF that can be evaluated in real-time to produce glinty effects.
- It is consistent with the microfacet BRDF of Cook and Torrance.
- It relies on a normalised patch-NDF which is evaluated in real-time from a pre-computed dictionary.
- The dictionary does not depend on roughness, allowing for spatial variations and roughness selection in real-time.
- Our model is compact (less than 1 MiB).

We first review the previous works on multiscale NDFs in Section 2. After a brief overview of our method (Section 3), we describe our multiscale high-frequency NDF in Section 4. Then we explain how to introduce spatial variations (Section 5) and present our reflectance model (Section 6). Results and comparisons are discussed in Section 7.

2. Previous works

The specular part of the BRDF is generally defined by three terms: 1) a micro-geometry model, i.e. microfacet distributions (its theoretical framework was introduced in Torrance and Sparrow [TS67]) usually represented by statistical normal distribution functions (NDFs), 2) a masking/shadowing term and 3) a Fresnel term. We focus on the first term and review single- and multi-scale NDFs. Note that we do not review the many other, mainly specific representations, that are outside the scope of microfacet theory like for example scratches [RGB16, VWH18, WVJH17], point jittering in 3D grids [WB16], bi-directional texture functions [GK17] or data-driven techniques.

Cook and Torrance [CT82] introduced a first physical BRDF to computer graphics using the Beckmann distribution [BS63]. It is based on a Gaussian parameterised by a roughness coefficient representing the distribution of microfacet slopes. Trowbridge and Reitz [TR75] proposed a distribution of slopes on an ellipsoid, later dubbed the GGX distribution by Walter et al. [WMLT07]. These models have a single-scale and cannot represent glitter. Multi-scale models are conversely able to represent sparkling surfaces. In the following, we distinguish models based on explicit normal maps and procedural algorithms.

Normal maps LEAN [OB10] and LEADR [DHI*13] mapping estimate the slope mean and variance within the pixel footprint and use them as parameters of a single lobe Beckmann NDF. The appearance of multi-scale smooth materials can be simulated, though the $\mathcal{P}$-NDF is only approximated. Han et al. [HSRG07, WZYR19] use a mixture of one-lobe von Mises-Fisher distributions to increase accuracy. However, the algorithmic complexity grows linearly with the number of lobes, so in practice there is a limitation to about a dozen lobes, which is still not sufficient to represent glittery materials. The $\mathcal{P}$-NDF [YHJ*14] aims at computing the exact NDF within a pixel footprint $\mathcal{P}$ more accurately. A 4D Bounded Volume Hierarchy (BVH), built from the normal map, allows for fast querying of light reflecting normals. Yan et al. [YHMR16] accurately approximate a normal map by a mixture of 4D NDFs to accelerate renderings. Mixtures of Beckmann distributions are used to improve BRDF normalisation [CCM18]. Extensions support multiple-scattering of lights within the normal map [CCM19], analytic light integration [GGN18] and diffraction effects [YHW*18, KHX*19]. Finally, Wang et al. [WHHY20] reduce memory consumption by using texture synthesis based on a normal map sample.

Procedural approach Explicitly storing very large normal maps might become an issue. Jakob et al. [JHY*14] generate microfacets at runtime while traversing a procedural 4D hierarchy. They calculate the ratio between the number of facets correctly oriented and the total number of microfacets inside $\mathcal{P}$. For all the previous cited methods, the accurate $\mathcal{P}$-NDF estimation entails high computation cost and thus none of them is applicable to real-time rendering. Some methods [AK16,WWH18] improve performance, but still not sufficiently to permit high performance. Conversely, Zirr and Kaplanyan [ZK16] reach real-time performance by making use of a MIP hierarchy instead of a complex data structure. During $\mathcal{P}$-NDF evaluation, they go through the footprint cells of the MIP hierarchy and, for each, evaluate the total number of microfacets using the cell area and a user-defined glint density. They also evaluate the number of reflecting microfacets in a cell by sampling a binomial distribution. For performance reasons, they approximate the probability of the binomial distribution, not allowing their $\mathcal{P}$-NDF to be normalised, leading to energy leaks and no physical validity. Although they achieve interesting glittery appearances, their specular lobe (Figure 2, right) does not converge to the lobe of Cook and Torrance (Figure 2, middle) when the microfacet density is very high. The normalisation of their BRDF is not straightforward because it depends on several parameters such as viewing direction, roughness and glint density. Our method also uses procedural NDFs and a MIP hierarchy. We replace the binomial sampling by a normalised NDF evaluation to guarantee a physically based BRDF. We thus avoid the previous issues (Figure 2, left), yet maintaining real-time rendering performance. Another real time physically-based stochastic glinty BRDF was recently proposed by Wang et al. [WDH20]. They can handle environment maps, contrary to our method. However, their memory footprint exceeds 256 MiB whereas our data requires only 384 KiB. They handle only isotropic roughness and cannot use roughness textures.

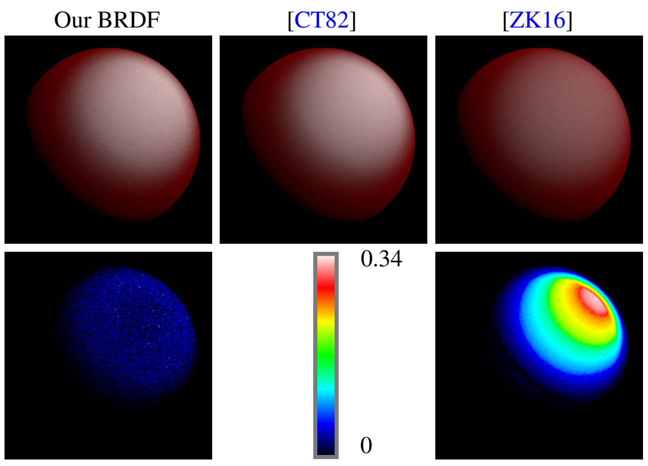

Figure 2: *The standard BRDF of Cook and Torrance [CT82] (middle) is the reference for an infinite number of microfacets. We compare the convergence of our model (left) and of the BRDF of Zirr and Kaplanyan [ZK16] (right), both with a very high number of microfacets. Top: rendering. Bottom: differences of radiance with the reference (range from* 0 *to the maximum* 0.34*, false-coloured).*

3. Overview

The main goal of our method is to efficiently render spatially dense high-frequency NDFs to reproduce the appearance of glinty materials. This is a challenging task because 2D high-frequency distributions are costly to store: naively, it requires to store high definition 4D fields, i.e. high-frequency NDFs on densely sampled surface points. We propose a solution to this problem by introducing a compact high-frequency NDF model, which can represent several dozens of fine lobes. We use the slope space domain to represent normals, which is common in microfacet theory. For example, both GGX and Beckmann NDFs are based on Slope Distributions Functions (SDFs). In this paper, all our NDFs are defined by SDFs.

Our model is illustrated in Figure 3:

(a) During rendering, we filter a *procedural MIP hierarchy*. Procedural meaning that each cell of the hierarchy does not explicitly store a SDF.
(b) Each cell index seeds a pseudo-random generator, which randomly picks *two 1D marginal distributions* from a *dictionary*.
(c) The product of the two 1D marginal distributions yields a *2D SDF*. Randomness avoids repetition artefacts and apparent periodicity. Moreover, we significantly reduce the memory footprint compared to explicit storage of SDFs in hierarchy cells.
(d) To further increase SDF variety, still using a small dictionary, a *random rotation* is applied to each SDF. It also avoids alignment artefacts.
(e) *Scaling* is applied to match a user-defined anisotropic roughness.
(f) A weighted sum over the cells covered by the footprint produces the $\mathcal{P}$-SDFs defined at a discrete LOD.
(g) Interpolation across the LOD results in the final multi-scale $\mathcal{P}$-SDF. The latter is linked with a microfacet-based BRDF to compute local shading.

We first describe our multiscale high-frequency SDF (Section 4). Next, we introduce the generation of spatial variations as well as the procedural MIP hierarchy (Section 5). Lastly, we derive the $\mathcal{P}$-SDF and the BRDF model (Section 6).

4. Multiscale high-frequency SDF

4.1. Preliminaries

The microsurface is characterised by a statistical distribution of the microfacets' orientations. It can be defined as a distribution of micronormals $\omega_m = (x_m, y_m, z_m)$ (m for micronormal), the normal distribution function $D(\omega_m)$ (NDF). Alternatively, it can be defined as a distribution of microfacets' slopes [Hei14]

$$\tilde{m} = (x_{\tilde{m}}, y_{\tilde{m}}) = \left(\frac{-x_m}{z_m}, \frac{-y_m}{z_m}\right), \tag{1}$$

the slope distribution function $P^{22}(\tilde{m})$ (SDF, $\tilde{m}$ means microslope). Both are related:

$$D(\omega_m) = \frac{P^{22}(\tilde{m})}{(\omega_m \cdot \omega_g)^4}, \tag{2}$$

where ω_g is the geometric normal of the surface (which should be the mean of the micronormals).

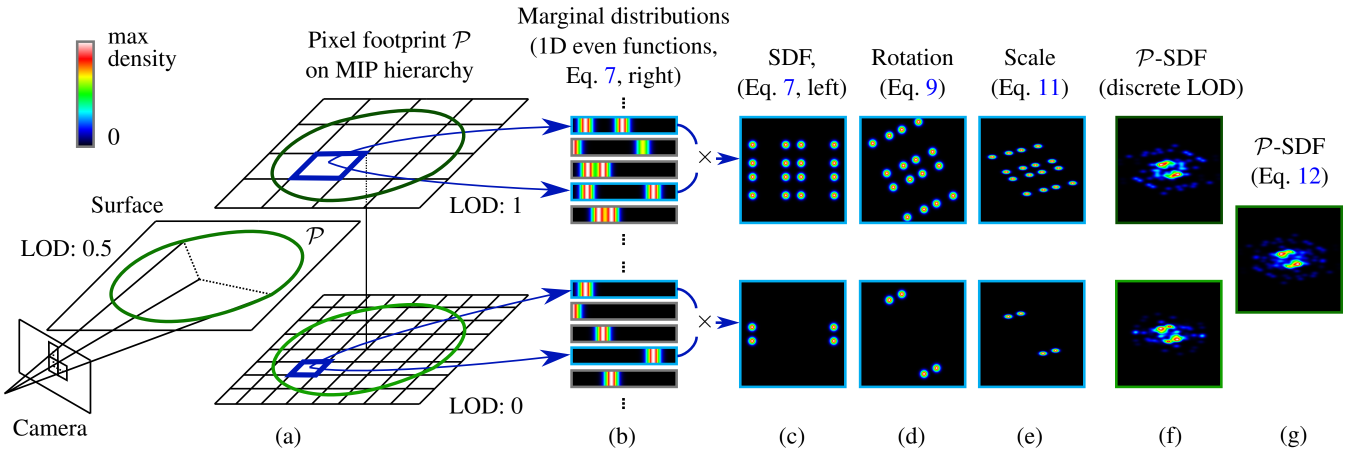
max density
0
Pixel footprint $\mathcal{P}$ on MIP hierarchy
Marginal distributions (1D even functions, Eq. 7, right)
SDF, (Eq. 7, left)
Rotation (Eq. 9)
Scale (Eq. 11)
$\mathcal{P}$-SDF (discrete LOD)
$\mathcal{P}$-SDF (Eq. 12)
Surface
LOD: 0.5
$\mathcal{P}$
LOD: 1
LOD: 0
Camera
(a)
(b)
(c)
(d)
(e)
(f)
(g)

Figure 3: *For each cell of a pixel footprint $\mathcal{P}$ on a MIP hierarchy (a) we randomly choose two 1D marginal distributions of slopes from a multiscale dictionary (b), whose product yields a SDF (c). The SDF is then randomly rotated to increase the output SDF variety of the dictionary (d). Finally, the SDF is scaled (e) to target a specified roughness. The weighted sum of the SDFs within $\mathcal{P}$ gives two $\mathcal{P}$-SDFs, defined for the two adjacent discrete LODs (f). By using the continuous LOD value, they are linearly interpolated, providing the $\mathcal{P}$-SDF (g) used in the microfacet BRDF.*

To be physically valid, it is mandatory that distributions are normalised. The projected area of the microsurface must be equal to the macrosurface area (set to unit area for convenience)

$$\int_{\Omega} (\omega_m \cdot \omega_g) D(\omega_m) \mathrm{d}\omega_m = 1, \tag{3}$$

which is equivalent by substitution to

$$\int_{\mathbb{R}^2} P^{22}(\tilde{m}) \mathrm{d}\tilde{m} = 1. \tag{4}$$

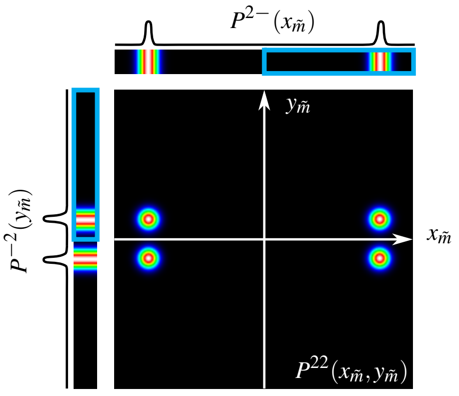

Figure 4: *Our 2D SDF is the product of two 1D marginal distributions for the slopes $x_{\tilde{m}}$ and $y_{\tilde{m}}$. The marginal distributions are even, so we only store the positive part of the function (blue frames).*

4.2. Single scale SDF

Our mathematical expression of P^{22} is an independent joint probability density function, which is the product of two univariate marginal distributions P^{2-} and P^{-2} along the x and y-axis:

$$P^{22}(\tilde{m}) = P^{2-}(x_{\tilde{m}}) P^{-2}(y_{\tilde{m}}). \tag{5}$$

A key property of this equation is that P^{22} is normalised if the marginal distributions are both normalised.

In order to avoid the paradoxical case where the micronormal mean is different from the geometric normal ω_g [DHI*13, SHHD17], we use symmetric, i.e. even, marginal density functions. This representation also allows us to halve storage, because in our implementation, these 1D distributions are tabulated. We illustrate Equation 5 and the symmetry of the distributions in Figure 4.

4.3. Multiscale SDF

In this Section we define and give the properties of our multiscale NDF, and we propose an algorithm to generate several.

Definition By introducing the discrete hierarchy LOD parameter l in equation 5, we define our 2D multiscale SDF

$$P^{22}(\tilde{m}, l) = P^{2-}(x_{\tilde{m}}, l) P^{-2}(y_{\tilde{m}}, l) \tag{6}$$

as the product of two 1D multiscale marginal distributions $P^{2-}(x_{\tilde{m}}, l)$ and $P^{-2}(y_{\tilde{m}}, l)$. The level ranges from $l = 0$ (finest scale) to *nLevels* (coarsest scale). In practice, we set *nLevels* $= 16$, since it experimentally represented a good compromise between compactness and visual quality.

Finest LOD In order to be able to model few microfacets, it is necessary for the finest LOD to have as few thin lobes as possible. Due to equation 5 and the use of even marginal distributions, we always have four lobes (Figure 4), even though pairs might overlap or coincide (in an extreme case, when they are centered around 0).

Coarsest LOD To make our model compliant with an existing microfacet BRDF, we let it converge to the Cook and Torrance reflectance model. This implies that P^{22} converges to a Beckmann distribution P^{22}_{target} as the number of microfacets increases. Therefore, we enforce $P^{22}(\tilde{m}, nLevels) = P^{22}_{\text{target}}(\tilde{m})$.

We chose the Beckmann SDF as target because the Gaussian is separable, i.e. it equals the product of its marginal distributions (Equation 5). If the target SDF is isotropic, we have $P^{2-}_{\text{target}} = P^{-2}_{\text{target}}$, which we denote as P_{target} for short. Distributions that are not separable are currently excluded (like the GGX distribution, see limitation discussion).

Transition between adjacent LODs Since we use a MIP hierarchy, the number of lobes quadruples when we move from level l to level $l+1$. Consequently, at level l, the multiscale SDF is a mixture of 4^{l+1} lobes and its marginal distributions are a mixture of 2^{l+1} lobes.

4.4. Generation algorithm

To generate a single multiscale SDF that enforces the previous properties, we have to define consistent lobes for P^{2-} and P^{-2} at all levels. We opted for Gaussian lobes with small standard deviation. The positions (means) of the Gaussian lobes are obtained by importance sampling of P_{target}: the first random sample defines the single lobe at $l = 0$, the second random sample is added for $l = 1$, etc. Figure 5 illustrates the algorithm and resulting convergence.

In our implementation, we fixed the following parameters, that produce glints while converging in a reasonable number of levels: the standard deviation of the Gaussian lobes is 0.02; the isotropic roughness of the target distribution is $\alpha_{\text{dict}} = 0.5$; the number of levels of details is $nLevels = 16$; the size of the tabulated generated monoscale SDFs is 64.

4.5. Dictionary of marginal distributions

To render glints, spatial variations over the surface are required and thus many different SDFs are necessary. We achieve this by building a dictionary of N marginal distributions P_i, as shown in Figure 6. The distributions are generated randomly with the previously described algorithm (Section 4.4). The dictionary can be arbitrarily large: the smaller N, the lower the memory footprint, whereas the larger N, the more variety is obtained.

To control roughness and anisotropy of P^{22}, we need to control the roughness of the target marginal distributions in x and y direction independently. Instead of having one dictionary per axis and one dictionary per roughness, which would require a lot of memory, we store a *single* dictionary, which is then rotated and scaled in real-time, as explained in Section 5. This strategy allows us to keep the memory footprint very low (less than 1 MiB).

5. Spatially-varying and multiscale SDF

To spatially vary the SDFs, we partition the surface into square cells and assign a SDF each of them. To manage the LODs, we use a MIP hierarchy, where each cell at level $l+1$ represents four adjacent cells at level l (Figure 3, a), as in a MIP pyramid. By using the 2D cell coordinate s_0, we can now define our spatially-varying and multiscale SDF:

$$P^{22}(\tilde{m}, l, s_0) = P^{2-}(x_{\tilde{m}}, l, s_0) P^{-2}(y_{\tilde{m}}, l, s_0). \tag{7}$$

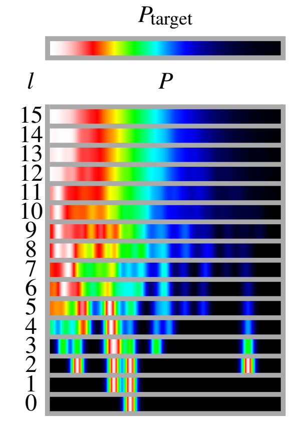

Figure 5: *The multiscale marginal distribution P (bottom) converges to a target distribution (top). Plots are restricted to $\mathbb{R}^+$ because we use even functions.*

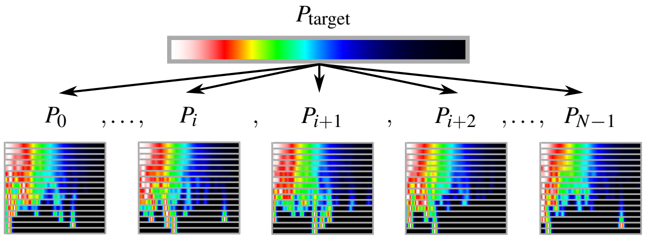
P_{target}
P_0 , . . . , P_i , P_{i+1} , P_{i+2} , . . . , P_{N-1}

Figure 6: *Dictionary of N multiscale marginal distributions. We generate a collection of multiscale distributions P_i that converge to the same target distribution P_{target}*

At this point, we have to maintain the coherence between levels *and* cell coordinates. The exact solution would be that the SDF of level $l+1$ equals the average of four adjacent SDFs of level l. However, it would require to precompute all the possible combinations in the dictionary, and lead to a combinatorial explosion as l grows. Instead, we use the same approximation as Zirr and Kaplanyan [ZK16]: a SDF at level $l+1$ retrieves the lobes of only *one* SDF at level l. The number of lobes quadruples, not due to the merging of four cells, but due to the change of level in the dictionary (Section 4.5).

5.1. Coherent indexing of cells

Equation 7 is evaluated by randomly picking P^{2-} and P^{-2} in the dictionary using s_0 as seed. To maintain the coherence at the transition from l to $l+1$, we need a coherent indexing, as shown in Figure 7. For given surface position s and level l, the cell coordinate is

$$s_0 = \left\lfloor \frac{s}{2^l} \right\rfloor 2^l. \tag{8}$$

Zirr and Kaplanyan [ZK16] also use coherent indexing, but for an infinite number of LODs.

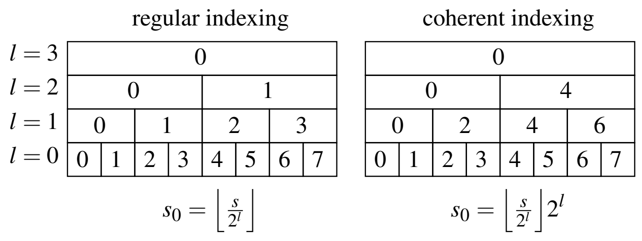

Figure 7: *Coherent cell indexing in the hierarchy. Conversely to the regular indexing (left), the coherent indexing (right) preserves one index out of four when transiting from l to $l+1$.*

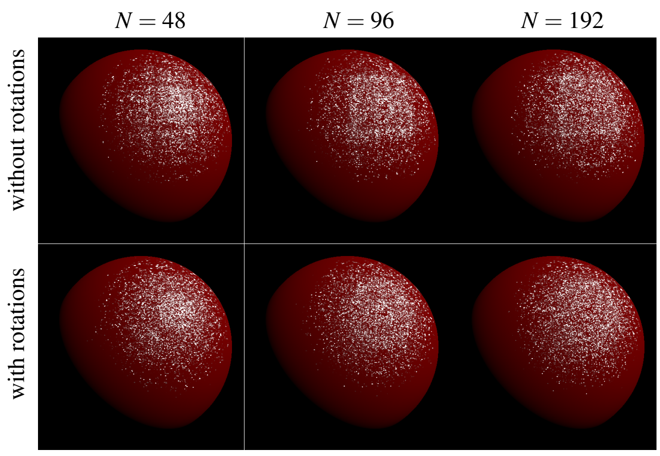

Figure 8: *Increasing the number N of marginal distributions in the dictionary reduces rendering artefacts, such as unrealistic low or high microfacet density along lines (top) or circles (bottom). Applying random rotations (bottom) to cell-dependent SDFs produces better results than SDFs without random rotations (top).*

5.2. Improving the variety by rotating the SDFs

Rich spatial variations are essential to avoid rendering artefacts, such as the apparent alignments in Figure 8 top left. A dictionary containing N marginal distributions can generate N^2 SDFs. We further increase the variety without additional storage by applying a random rotation R_θ to the SDF: a random angle θ uniformly distributed in $[0; 2\pi[$ is generated from the cell index s_0. The result on a single SDF is illustrated in Figure 3 (d). The result on the surface is illustrated in Figure 8 bottom: axis-aligned artefacts are removed; slight circle artefacts appear but the quality is dramatically improved for the same dictionary size N. The rotated SDF is evaluated by inverse rotation on the slope parameter:

$$P^{22}_{\theta}(\tilde{m}, l, s_0) = P^{22}(R^{-1}_{\theta}\tilde{m}, l, s_0). \tag{9}$$

Note that the rotation preserves the normalisation (Equation 4). If the target distribution P^{22}_{target} is isotropic, then P^{22}_{θ} also converges to P^{22}_{target}. We have been inspired by the area light integration method, which uses linearly transformed distributions [HDHN16]. Rotations have also been used to match BTF data [WDR11].

5.3. Controlling the roughness by scaling the SDFs

Controlling the roughness is necessary to adjust the material appearance. We allow the user to specify an arbitrary roughness $\alpha = (\alpha_x, \alpha_y)$ of an anisotropic Beckmann distribution (target). As in the previous section, we linearly transform the slope parameter instead of the functions, which allows to keep a single dictionary. Let α_{dict} be the target roughness the dictionary has been built with. Let the scaling matrix be:

$$S_\alpha = \frac{1}{\alpha_{\text{dict}}} \begin{bmatrix} \alpha_x & 0 \\ 0 & \alpha_y \end{bmatrix}. \tag{10}$$

We define our complete SDF model as

$$P^{22}_{\theta\alpha}(\tilde{m}, l, s_0) = \frac{\alpha^2_{\text{dict}}}{\alpha_x \alpha_y} P^{22}(S^{-1}_{\alpha} R^{-1}_{\theta} \tilde{m}, l, s_0) \tag{11}$$

which is spatially-varying (s_0), multiscale (l), rotated (θ), and controllable with anisotropic roughness (α). The normalisation factor preceeding P^{22} is the Jacobian of S_α, which ensures Equation 4 (normalisation) to remain fulfilled.

6. Reflectance model

The microfacet theory allows us to define a physically based reflectance model from a $\mathcal{P}$-SDF. We will defined it before introducing the definition of our multiscale BRDF.

6.1. Patch-SDF based on a MIP hierarchy

The patch $\mathcal{P}$ refers to the pixel footprint (green ellipse Figure 3, a) or the ray footprint. To approximate the real footprint of a pixel with a parallelogram or a Gaussian, ray differentials are commonly used [Ige99]. In our work, we use a Gaussian representation $W_{\mathcal{P}}$ [Hec89], but other weighting functions are also valid with our method.

When using a MIP hierarchy in combination with our scaled, rotated, spatially-varying and multiscale SDFs, the $\mathcal{P}$-SDF is simply the result of the filtering of the MIP hierarchy:

$$P^{22}_{\mathcal{P}}(\tilde{m}) = \sum_{l=\lfloor l_{\mathcal{P}} \rfloor}^{\lfloor l_{\mathcal{P}} \rfloor + 1} w(l) \sum_{s_0 \in \mathcal{P}} P^{22}_{\theta\alpha}(\tilde{m}, l, s_0) W_{\mathcal{P}}(l, s_0), \tag{12}$$

where $l_{\mathcal{P}}$ is the continuous LOD associated to $\mathcal{P}$. The weighting functions are normalised, i.e. $w(\lfloor l_{\mathcal{P}} \rfloor) + w(\lfloor l_{\mathcal{P}} \rfloor + 1) = 1$ for the levels and $\sum_{s_0 \in \mathcal{P}} W_{\mathcal{P}}(s_0, l) = 1$ for the positions. As for the case without patch (Equation 2), we use the slope-to-normal transformation to go back to normal space, and obtain the $\mathcal{P}$-NDF:

$$D_{\mathcal{P}}(\omega_m) = \frac{P^{22}_{\mathcal{P}}(\tilde{m})}{(\omega_m \cdot \omega_g)^4}. \tag{13}$$

Our definition (Equation 12) is normalised because the distributions $P^{22}_{\theta\alpha}(\tilde{m}, l, s_0)$ integrate to one over the slope domain.

6.2. Multiscale microfacet based BRDF

The patch-BRDF $f_{\mathcal{P}}$ of the pixel footprint's material, characterised by $D_{\mathcal{P}}$, is

$$f_{\mathcal{P}}(\omega_o,\omega_i) = \frac{F(\omega_o,\omega_h)G_2(\omega_o,\omega_i,\omega_h)D_{\mathcal{P}}(\omega_h)}{4(\omega_o\cdot\omega_g)(\omega_i\cdot\omega_g)}, \tag{14}$$

where ω_o and ω_i are the observation and incident directions, and ω_g is the geometric normal. The term ω_h refers to the half vector of reflection, F is the Fresnel factor, and G_2 is the masking-shadowing function. Our $\mathcal{P}$-NDF does not have analytic Smith masking-shadowing function [Smi67], so we use the V-cavity masking-shadowing function [CT82] which does not rely on the shape of the $\mathcal{P}$-NDF:

$$G_2(\omega_o,\omega_i,\omega_m) = G_1(\omega_o,\omega_m)G_1(\omega_i,\omega_m) \tag{15}$$

with masking $G_1(\omega_o,\omega_m)$ and shadowing $G_1(\omega_i,\omega_m)$. The V-cavity masking function is

$$G_1(\omega,\omega_m) = H(\omega\cdot\omega_m)\min\left(1, 2\frac{(\omega_m\cdot\omega_g)(\omega\cdot\omega_g)}{(\omega\cdot\omega_m)}\right) \tag{16}$$

where H is the Heaviside function.

7. Results

In this section, we describe the user parameters and the practical implementation of our BRDF. We also evaluate its results, quality and performance, and discuss the limitations of our model.

7.1. User parameters

An artist can control the glinty material appearance by changing the following parameters:

- *Roughness* α_x and α_y control the size of the area where almost all glints are located on the surface. Our multiscale BRDF converges to the BRDF of Cook and Torrance with α_x and α_y.
- *Microfacet density* ρ is the number of specular microfacets per unit surface area. Material appearance is glintier with small values and smoother with high values.
- *Microfacet relative area* β gives the percentage of the surface covered by our glinty BRDF. Set to one this yields rough plastics or metals, while small values yield materials such as stone with sparkling minerals. Note that β acts as a mask on the surface, and it is independent of ρ.

Please see the supplementary video for the demonstration of the real-time user control of these parameters.

We also provide more technical parameters:

- *Density randomisation* ζ randomly changes the density of microfacets around ρ. These random variations reduce rendering artefacts when the number N of marginal distributions in the dictionary is low. Zirr and Kaplanyan [ZK16] also use a similar parameter.
- *Maximum anisotropy* γ limits the ratio between major and minor axis of the pixel footprint and thus the number of cells to go through during $\mathcal{P}$-SDF evaluation. The specular lobes are respectively coarse or fine according to low or high values.

7.2. Validation of our model

To validate our patch microfacet BRDF $f_{\mathcal{P}}$, we use the weak white furnace test [Hei14]. Without the shadowing $G_1(\omega_i,\omega_m)$ and Fresnel F terms, our BRDF integrates to one:

$$\int_{\Omega} \frac{G_1(\omega_o,\omega_h)D_{\mathcal{P}}(\omega_h)}{4(\omega_g\cdot\omega_o)}\mathrm{d}\omega_i = 1, \tag{17}$$

because our $\mathcal{P}$-NDF is normalised (i.e., fulfills Equation 3), averages to the geometric normal ω_g and is symmetrical as we use even density functions. This test guarantees that our BRDF does not create energy and is, therefore, physically based. Figure 9 shows numerical integration of equation 17 for various values of parameters α, ρ, ζ and ω_o: the relative error is below 0.003, except for a very low α at grazing angle (0.012 error).

Note that we do not use the non-weak white furnace test, as we only model the first scattering event; the rays that bounce multiple times on the microsurface are removed from the BRDF by the shadowing function. Our formulation also respects the Helmholtz reciprocity, i.e., $f_{\mathcal{P}}(\omega_o,\omega_i) = f_{\mathcal{P}}(\omega_i,\omega_o)$.

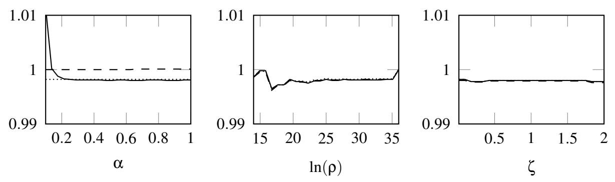

Figure 9: *Numerical validation of our BRDF with the weak white furnace test. Equation 17 is plotted against different parameters. Different observation polar angles θ_o (between ω_o and ω_g) are shown: dotted line for $\theta_o = 0$; dashed line for $\theta_o = \pi/4$; solid line for $\theta_o = 1.56$ (grazing angle).*

7.3. Implementation

We have implemented our method in OpenGL 4.5 and in pbrt-v3 [PJH16] to compare it with an offline and photo-realistic method. In both cases, the implementation is similar. We provide pseudo-code and implementation details in appendix A. Our supplemental materials contain our implementation and the dictionary generator.

Dictionary The marginal distributions P_i of the dictionary (Figure 6) are stored in 1D tables of size 64, which provides enough precision to represent several distinct narrow lobes. The standard deviation of the lobes is set to 0.02 in our implementation. The domain of the 1D tables is $[0, 4\alpha_{\text{dict}}/\sqrt{2}]$, which corresponds to four times the standard deviation of P^{22}_{target}, allowing the representation of almost all the density of the distribution. All our marginal distributions are generated and normalised assuming linear interpolation between discrete values. In all our scenes, we use $N = 192$ marginal distributions in the single dictionary, leading to artefact free renderings (Figure 8). Generating and normalising 192 distributions takes 30 s multi-threaded on a 2.20 GHz Intel Core i7-8750H CPU (12 threads).

7.4. Renderings

Sponza In Figure 1, left, all the materials use our glinty multiscale BRDF, with different parameters. The floor, the columns, the arches and the vases are rough rocks composed of a low proportion of sparkling minerals. In the real world, this material has strong flickers at high zoom levels and little or no flicker at low zoom levels, as in our rendering. The sparkling fabrics are less rough and have a larger microfacet relative area. The metallic curtain rods produce no glint because the microfacet density is high.

Screen Plane In Figure 1, right, we show that microfacet densities can vary along the surface and that our BRDF converges to the BRDF of Cook and Torrance. The roughness is isotropic and the microfacet logarithmic density linearly increases from left to right.

Suzanne In the off-line BRDF of Yan et al. [YHMR16], the $\mathcal{P}$-NDF is normalised, as in our method. Consequently, we achieve similar results (shown in Figure 10), where glint intensities are similar. Since the $\mathcal{P}$-NDF of Zirr and Kaplanyan [ZK16] lacks normalisation, the appearance is darker in specular regions. This scene also shows that our method can handle anisotropic roughness and simulate the appearance of rough plastics.

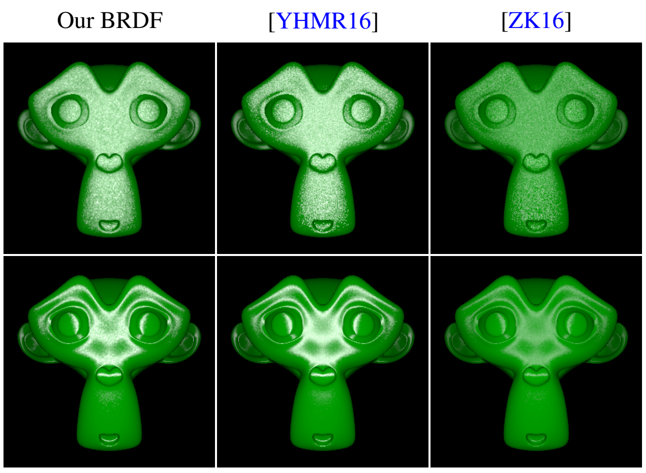

Figure 10: *Comparison with an offline physically-based model. Suzanne is rendered with different microfacet BRDFs with the same roughness and specular parameters. Our model (left) has glint intensities similar to those of Yan et al. [YHMR16] (middle), whereas the method of Zirr and Kaplanyan [ZK16] gives darker appearances (right). We handle isotropic (top) and anisotropic (bottom) surface roughness.*

2 CV In Figure 11, we demonstrate that non-uniform glint colouration can be modelled with our BRDF to simulate the appearance of car paint. For each cell index s_0, we randomly pick a colour out of a user-defined table and we multiply it by the SDF. In this scene, we mix our BRDF with the BRDF of Cook and Torrance. The former simulates the metallic flakes, the latter models the transparent varnished of the car paint. The two BRDFs combine well, and the mixture gives a realistic result, without post-processing, because our BRDF is physically based.

Parquet Since our BRDF handles arbitrary roughness on-the-fly, by scaling the distributions (Section 5.3), we can use roughness maps as shown in Figure 12. The roughness map modulates the shape of the highlights, allowing non elliptical specular lobes. In this scene, we use three point lights with different intensities.

Brushed aluminium Stretching of the microsurface allows for anisotropic glints, as demonstrate in Figure 13. Brushed metal with elongated grooves can therefore be modelled. In our implementation, we stretch the surface position $s = (x_s, y_s)$ by 1,000 along x_s and set the maximum anisotropy γ of the pixel footprint to 8.

Table 1 details the parameters used in our scenes. The maximum anisotropy of $\mathcal{P}$ is always set to 4, except in the Brushed Aluminium case. There is no post-effect applied to the rendering. Please see the temporal versions of the Sponza, 2 CV and Parquet scenes in the attached video.

Scene	α_x	α_y	$\ln(\rho)$	β	ζ
Sponza (stones)	1	1	20	0.01	2
Sponza (fabrics)	0.3	0.3	20	0.2	2
Sponza (metal)	0.1	0.1	26	1	2
Screen plane	0.5	0.5	$[20,35]$	1	2
Suzanne (top)	0.5	0.5	21	1	0.01
Suzanne (bottom)	0.5	0.1	21	1	0.01
2 CV	0.1	0.1	23	0.3	0.01
Parquet	$T(s)$	$T(s)$	21	1	0.01
Brushed alu.	0.5	0.1	20.6	0.9	0.01

Table 1: *Parameters used in our scenes. The notation $[a,b]$ means that the parameter is linearly interpolated along the surface between a and b. The notation $T(s)$ denotes textured values.*

Scene	#Triangles	Our BRDF	[ZK16]	[CT82]
Sponza	262,267	3.0	2.0	0.7
Screen plane	2	2.5	1.3	0.4
2 CV	704,527	9.2	4.4	2.0
Parquet	2	6.4	2.5	0.6
Brushed alu.	16,128	1.7	0.7	0.4

Table 2: *Rendering times in milliseconds per frame using our OpenGL implementation. We compare the performance of our BRDF, the BRDF of Zirr and Kaplanyan [ZK16], and the BRDF of Cook and Torrance [CT82].*

7.5. Performance and memory requirement

Performance The average rendering times (ms/frame) of the different scenes are summarized in Table 2, with our BRDF evaluated using an NVIDIA GeForce 2080 RTX GPU. For comparisons, we also give timings for the BRDF of Zirr and Kaplanyan [ZK16] and Cook and Torrance [CT82]. Measurements are made for 1920 × 1080 image resolution using forward shading (not deferred shading). Consequently, the rendering times include shading of occluded pixels due to overdraw in large scenes. Rendering times increase with the screen coverage of glints, which itself increases with the relative microfacet area β and roughness values α.

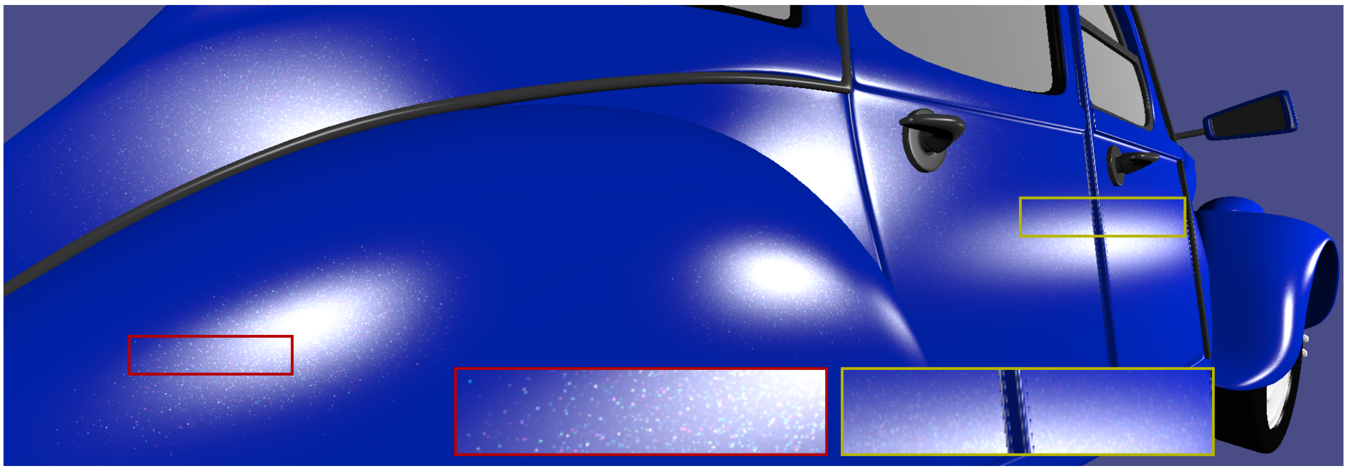

Figure 11: *A 2 CV is rendered with our BRDF using coloured glints to simulate car paint.*

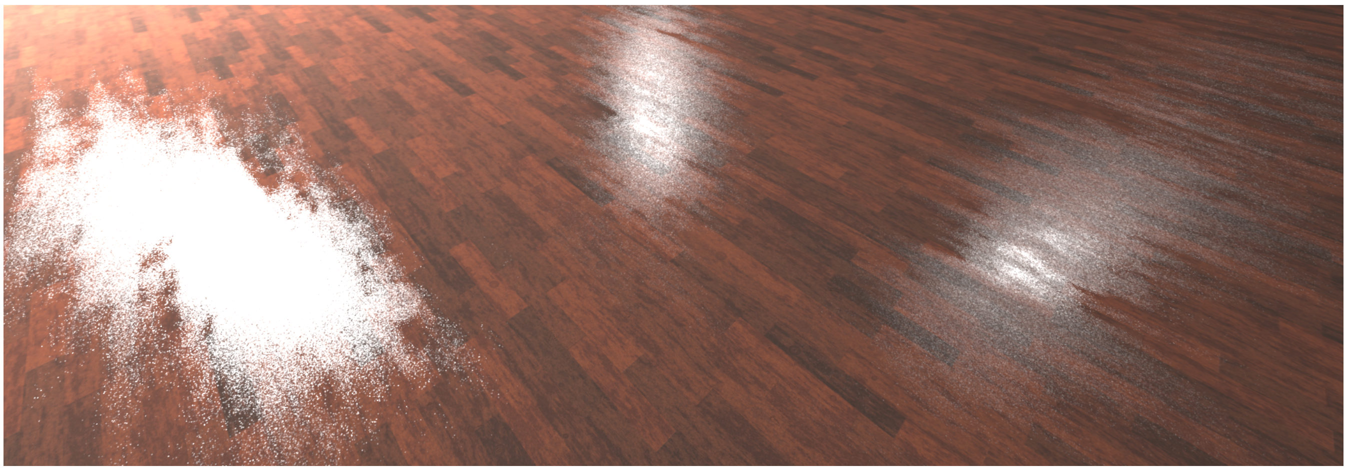

Figure 12: *A varnished parquet made from dark wood is rendered with our glinty BRDF. In this scene, we use three point lights with three different intensities and also a roughness map to modulate the highlights along the surface.*

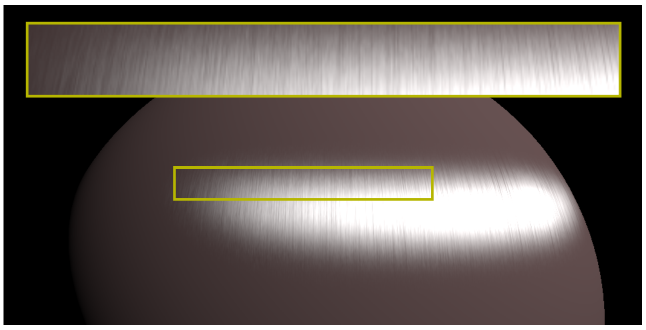

Figure 13: *The appearance of brushed aluminium can be modelled with our BRDF by stretching the surface position.*

In the 2 CV and Parquet scenes, rendering times are more important than the other scenes because we use three point lights instead of one. Compared to Zirr and Kaplanyan, the rendering time overhead varies from 50 % to about 156 % in the worst case. Our tests show that the memory traffic between the compute units and the cache causes the performance reduction.

Memory requirement As described in Section 4.5, we store a dictionary in memory. For this purpose, we allocate a single OpenGL 1D array texture with 3-component half-float format (16 bits per component). Normalised integers with 8 bit per component cannot handle the large range of values of the distributions, whereas half-float precision is enough. For 16 levels and 192 marginal distributions, the number of 1D texture is 1,024, with three distributions per texture. Consequently, the memory requirement of our BRDF is 384 KiB.

7.6. Limitations and future works

To be physically based, our BRDF cannot use the Smith model. However, if the V-Cavity is not the standard model in the rendering engine, the Smith masking-shadowing function of the target distribution can be used without resulting in major rendering artefacts. The approximated SDF coherence between grid levels results in progressive introduction of new glints when zooming in and out. In most situations, this is not really noticeable, except at high zoom levels.

Not all distributions can be represented by our model. The GGX distribution is not supported, and we will investigate this limitation in a future work by looking for a good approximation, which fits our separable model. In addition, the anisotropic SDFs are always axis-aligned. Another limitation is that $\mathcal{P}$-NDFs cannot be controlled by normal maps or other explicit surface models.

We only model single scattering of light within the microsurface. Pre-computed normalisation factors of BRDFs [CK17, Tur19] cannot be used because we should store one factor per roughness *and* per marginal distribution pair in the dictionary (N^2 pairs). Finally, we did not yet propose an importance sampling scheme of our procedural NDF that we leave to future work. Therefore, area and environment maps are not supported yet.

8. Conclusion

We have proposed a novel physically based BRDF designed for real-time rendering of glints. To the best of our knowledge, no other method unifies, as we do, compactness, real-time performance and physical validity. Our method is procedural and uses a single and compact dictionary independent of the surface roughness. Glinty appearances can be reproduced, such as sparkling rock or fabric, rough metal and plastic, car paint and brushed metal. The reflectance model parameters can be controlled in real-time or varied across the surface, e.g. by using textures. Our BRDF matches well with the microfacet theory and is consistent with the BRDF of Cook and Torrance, and its physical validation is proven by the weak white furnace test. We have also provided a detailed pseudocode that allows implementing the theoretical model.

Acknowledgements

This work has been published under the framework of the IdEx Unistra supported by the Investments for the future program of the french government. This work has been partially funded by the project ReProcTex from the Agence Nationale de la Recherche (ANR-19-CE33-0011-01). Sponza scene is provided by Morgan McGuire (https://casual-effects.com/data) and 2 CV model is provided by David Norman (http://www.blendswap.com/blends/view/46830).

References

[AK16] Atanasov A., Koylazov V.: A Practical Stochastic Algorithm for Rendering Mirror-like Flakes. In *ACM SIGGRAPH Talks* (2016). 3

[BS63] Beckmann P., Spizzichino A.: *The scattering of electromagnetic waves from rough surfaces*. Pergamon Press, 1963. 2

[CCM18] Chermain X., Claux F., Mérillou S.: A microfacet-based brdf for the accurate and efficient rendering of high-definition specular normal maps. *The Visual Computer* (2018). 2

[CCM19] Chermain X., Claux F., Mérillou S.: Glint Rendering based on a Multiple-Scattering Patch BRDF. *Comput. Graph. Forum (Proc. Eurographics Symposium on Rendering) 38*, 4 (2019), 27–37. 2

[CK17] Conty A., Kulla C.: Revisiting physically based shading at Imageworks. In *ACM SIGGRAPH Courses* (2017), pp. 7:1–7:8. 10

[CT82] Cook R. L., Torrance K. E.: A Reflectance Model for Computer Graphics. *Comput. Graph. (Proc. SIGGRAPH) 1*, 1 (1982), 7–24. 1, 2, 3, 7, 8

[DHI*13] Dupuy J., Heitz E., Iehl J.-C., Poulin P., Neyret F., Ostromoukhov V.: Linear Efficient Antialiased Displacement and Reflectance Mapping. *ACM Trans. Graph. 32*, 6 (2013), 211:1–211:11. 2, 4

[GGN18] Gamboa L. E., Guertin J.-P., Nowrouzezahrai D.: Scalable Appearance Filtering for Complex Lighting Effects. *ACM Trans. Graph. 37*, 6 (2018), 277:1–277:13. 2

[GK17] Golla T., Klein R.: An Efficient Statistical Data Representation for Real-Time Rendering of Metallic Effect Car Paints. In *Virtual Reality and Augmented Reality* (2017), pp. 51–68. 2

[HDHN16] Heitz E., Dupuy J., Hill S., Neubelt D.: Real-Time Polygonal-Light Shading with Linearly Transformed Cosines. *ACM Trans. Graph. (Proc. SIGGRAPH) 35*, 4 (2016). 6

[Hec89] Heckbert P. S.: *Fundamentals of Texture Mapping and Image Warping*. Master's thesis, University of California, Berkeley, 1989. 6

[Hei14] Heitz E.: Understanding the Masking-Shadowing Function in Microfacet-Based BRDFs. *Journal of Computer Graphics Techniques 3*, 2 (2014), 48–107. 3, 7

[HSRG07] Han C., Sun B., Ramamoorthi R., Grinspun E.: Frequency Domain Normal Map Filtering. *ACM Trans. Graph. (Proc. SIGGRAPH) 26*, 3 (2007). 2

[Ige99] Igehy H.: Tracing ray differentials. In *Proc. ACM SIGGRAPH* (1999), p. 179–186. 6

[JHY*14] Jakob W., Hašan M., Yan L.-Q., Lawrence J., Ramamoorthi R., Marschner S.: Discrete Stochastic Microfacet Models. *ACM Trans. Graph. (Proc. SIGGRAPH) 33*, 4 (2014). 3

[KHX*19] Kuznetsov A., Hašan M., Xu Z., Yan L.-Q., Walter B., Kalantari N. K., Marschner S., Ramamoorthi R.: Learning Generative Models for Rendering Specular Microgeometry. *ACM Trans. Graph. 38*, 6 (2019). 2

[OB10] Olano M., Baker D.: LEAN Mapping. In *Proc. ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (2010), pp. 181–188. 2

[PJH16] Pharr M., Jakob W., Humphreys G.: *Physically Based Rendering: From Theory to Implementation*, 3rd ed. Morgan Kaufmann Publishers Inc., 2016. 7

[RGB16] Raymond B., Guennebaud G., Barla P.: Multi-scale Rendering of Scratched Materials Using a Structured SV-BRDF Model. *ACM Trans. Graph. (Proc. SIGGRAPH) 35*, 4 (2016), 57:1–57:11. 2

[SHHD17] Schüssler V., Heitz E., Hanika J., Dachsbacher C.: Microfacet-Based Normal Mapping for Robust Monte Carlo Path Tracing. *ACM Trans. Graph. (Proc. SIGGRAPH Asia) 36*, 6 (2017). 4

[Smi67] Smith B.: Geometrical shadowing of a random rough surface. *IEEE Transactions on Antennas and Propagation 15*, 5 (1967), 668–671. 7

[TR75] Trowbridge T. S., Reitz K. P.: Average irregularity representation of a rough surface for ray reflection. *Journal of the Optical Society of America 65*, 5 (1975), 531–536. 2

[TS67] Torrance K. E., Sparrow E. M.: Theory for Off-Specular Reflection From Roughened Surfaces. *Journal of the Optical Society of America 57*, 9 (1967), 1105–1114. 2

[Tur19] TURQUIN E.: *Practical multiple scattering compensation for microfacet models*. Tech. rep., Industrial Light & Magic, 2019. 10

[VWH18] VELINOV Z., WERNER S., HULLIN M. B.: Real-Time Rendering of Wave-Optical Effects on Scratched Surfaces. *Comput. Graph. Forum (Proc. of Eurographics) 37*, 2 (2018), 123–134. 2

[WB16] WANG B., BOWLES H.: A Robust and Flexible Real-Time Sparkle Effect. In *Proc. Eurographics Symposium on Rendering* (2016), p. 49–54. 2

[WDH20] WANG B., DENG H., HOLZSCHUCH N.: Real-Time Glints Rendering With Pre-Filtered Discrete Stochastic Microfacets. *Comput. Graph. Forum* (2020). 3

[WDR11] WU H., DORSEY J., RUSHMEIER H.: A Sparse Parametric Mixture Model for BTF Compression, Editing and Rendering. *Comput. Graph. Forum (Proc. of Eurographics) 30*, 2 (2011), 465–473. 6

[WHHY20] WANG B., HAŠAN M., HOLZSCHUCH N., YAN L.-Q.: Example-based microstructure rendering with constant storage. *ACM Trans. Graph. (Proc. SIGGRAPH) 39*, 5 (2020). 2

[WMLT07] WALTER B., MARSCHNER S. R., LI H., TORRANCE K. E.: Microfacet Models for Refraction Through Rough Surfaces. In *Proc. Eurographics Symposium on Rendering* (2007), pp. 195–206. 2

[WVJH17] WERNER S., VELINOV Z., JAKOB W., HULLIN M. B.: Scratch Iridescence: Wave-Optical Rendering of Diffractive Surface Structure. *ACM Trans. Graph. (Proc. SIGGRAPH) 36*, 6 (2017). 2

[WWH18] WANG B., WANG L., HOLZSCHUCH N.: Fast Global Illumination with Discrete Stochastic Microfacets Using a Filterable Model. *Comput. Graph. Forum (Proc. Pacific Graphics) 37*, 7 (2018), 55–64. 3

[WZYR19] WU L., ZHAO S., YAN L.-Q., RAMAMOORTHI R.: Accurate Appearance Preserving Prefiltering for Rendering Displacement-Mapped Surfaces. *ACM Trans. Graph. (Proc. SIGGRAPH) 38*, 4 (2019). 2

[YHJ*14] YAN L.-Q., HAŠAN M., JAKOB W., LAWRENCE J., MARSCHNER S., RAMAMOORTHI R.: Rendering Glints on High-Resolution Normal-Mapped Specular Surfaces. *ACM Trans. Graph. (Proc. SIGGRAPH) 33*, 4 (2014). 2

[YHMR16] YAN L.-Q., HAŠAN M., MARSCHNER S., RAMAMOORTHI R.: Position-Normal Distributions for Efficient Rendering of Specular Microstructure. *ACM Trans. Graph. (Proc. SIGGRAPH) 35*, 4 (2016). 2, 8

[YHW*18] YAN L.-Q., HAŠAN M., WALTER B., MARSCHNER S., RAMAMOORTHI R.: Rendering Specular Microgeometry with Wave Optics. *ACM Trans. Graph. (Proc. SIGGRAPH) 37*, 4 (2018). 2

[ZK16] ZIRR T., KAPLANYAN A. S.: Real-time Rendering of Procedural Multiscale Materials. In *Proc. ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (2016), pp. 139–148. 1, 3, 5, 7, 8

Appendix A: Implementation

We provide a pseudo-code with practical details. Algorithm 1 evaluates our glinty BRDF by computing the pixel footprint $\mathcal{P}$, the LOD and the $\mathcal{P}$-SDF. The discrete $\mathcal{P}$-SDF (Algorithm 2) averages the SDFs (Algorithm 3). Most parameters defined in Section 7.1 are used in Algorithm 3:

- The microfacet density ρ allows the computation of a distribution LOD l_{dist} which replaces the hierarchy LOD l in Equation 11. We first calculate the target number of microfacets n in a cell with the discrete LOD $\lfloor l \rfloor$ and the microfacet density ρ (Alg. 3, line 5). We know that the current number of lobes (microfacets) at level l is 4^{l+1}, as we have four lobes at LOD zero. We can deduce that $l_{\text{dist}} = \frac{\log(n)}{2\log(2)}$ is the distribution level corresponding to the user-defined microfacet density (Alg. 3, line 6).
- We use the microfacet relative area β in combination with a uniform random number $\mathcal{U}_\beta$ to discard cells (Alg. 3, line 4).
- The density randomisation ζ is the standard deviation of a normal distribution $\mathcal{N}(l_{\text{dist}}, \zeta^2)$, centred around the distribution LOD l_{dist}. We sample the normal distribution using a pseudo random number $\mathcal{U}_\zeta$ to get a randomised l_{dist} (Alg. 3, line 8).
- Finally, we can introduce glint colour variations by using a colour table with N_c colours (Alg. 3, lines 12 and 19).

Algorithm 1: Evaluation of $f_{\mathcal{P}}(\omega_o, \omega_i)$ (Eq. 14)

```
1  $\omega_h \leftarrow \frac{\omega_o+\omega_i}{||\omega_o+\omega_i||}$
2  $\tilde{h} \leftarrow \text{normalToSlope}(\omega_h)$                      // Eq. 1
3  $\mathcal{P} \leftarrow \text{computePixelFootprint}()$
4  $\mathcal{P} \leftarrow \text{clamp}(\mathcal{P}, \gamma)$        // Clamp pixel footprint eccentricity
5  $\text{minorLength} \leftarrow \text{minorLength}(\mathcal{P})$       // Get $\mathcal{P}$ minor length
6  $l \leftarrow \max(0, nLevels - 1 + \text{log2}(\text{minorLength}))$   // Get LOD
7  $w \leftarrow l - \lfloor l \rfloor$                                 // Get LOD weight
8  $P^{22}_{\mathcal{P}}(\tilde{h}) \leftarrow \text{lerp}(P^{22}_{\lfloor\mathcal{P}\rfloor}(\lfloor l \rfloor, \tilde{h}), P^{22}_{\lfloor\mathcal{P}\rfloor}(\lfloor l \rfloor + 1, \tilde{h}), w)$   // Eq. 12
9                                                          // Calls Alg 2
10 $D_{\mathcal{P}}(\omega_h) \leftarrow \frac{P^{22}_{\mathcal{P}}(\tilde{h})}{(\omega_h \cdot \omega_g)^4}$                 // Eq. 13
11 $f_{\mathcal{P}}(\omega_o, \omega_i) \leftarrow \frac{F(\omega_o,\omega_h)G_2(\omega_o,\omega_i,\omega_h)D_{\mathcal{P}}(\omega_h)}{4(\omega_o\cdot\omega_g)(\omega_i\cdot\omega_g)}$
```

Algorithm 2: $P^{22}_{\lfloor\mathcal{P}\rfloor}(\lfloor l \rfloor, \tilde{h})$ – $\mathcal{P}$-SDF for a discrete LOD

```
1 $P^{22}_{\lfloor\mathcal{P}\rfloor} \leftarrow 0$
2 foreach $s_0 \in \mathcal{P}$ do
3     $P^{22}_{\lfloor\mathcal{P}\rfloor} \leftarrow P^{22}_{\lfloor\mathcal{P}\rfloor} + P^{22}_{\theta\alpha}(\tilde{h}, l, s_0) W_{\mathcal{P}}(\lfloor l \rfloor, s_0)$   // Calls Alg. 3
4 end
```

Algorithm 3: $P^{22}_{\theta\alpha}(\tilde{h}, l, s_0)$ (Eq. 11)

```
1  $s_0 \leftarrow s_0 2^l$                          // Coherent indexing (Eq. 8)
2  $\text{rng.Seed}(s_0)$                  // Seed pseudo random generator
3  $\mathcal{U}_\beta \leftarrow \text{rng.UniformFloat}()$
4  if $\mathcal{U}_\beta > \beta$ then return 0
5  $n \leftarrow \frac{2^{2\lfloor l \rfloor}}{2^{2(nLevels-1)}}\rho$          // Number of microfacets in a cell
6  $l_{\text{dist}} \leftarrow \frac{\log(n)}{2\log(2)}$                // Continuous distribution LOD
7  $\mathcal{U}_\zeta \leftarrow \text{rng.UniformFloat}()$
8  $l_{\text{dist}} \leftarrow \text{sample}(\mathcal{U}_\zeta, \mathcal{N}(l_{\text{dist}}, \zeta^2))$      // Density randomisation
9  $l_{\text{dist}} \leftarrow \text{clamp}(\text{round}(l_{\text{dist}}), 0, nLevels)$    // Clamp dist. LOD
10 if $l_{\text{dist}} = nLevels$ then return $P^{22}_{\text{target}}(\tilde{h}, \alpha_x, \alpha_y)$
11 $\mathcal{U}_c \leftarrow \text{rng.UniformFloat}()$
12 $C \leftarrow \text{colourTable}[N_C \mathcal{U}_c]$                  // Coloured glints
13 $\mathcal{U}_r \leftarrow \text{rng.UniformFloat}()$
14 $R_\theta^{-1} \leftarrow \text{computeRotationMatrix}(\theta \leftarrow 2\pi\mathcal{U}_r)$
15 $S_\alpha^{-1} \leftarrow \text{computeScaleMatrix}(\alpha_x, \alpha_y, \alpha_{\text{dict}})$          // Eq. 10
16 $\tilde{h} \leftarrow S_\alpha^{-1} R_\theta^{-1} \tilde{h}$          // Scale and rotation of the slope $\tilde{h}$
17 $(\mathcal{U}_1, \mathcal{U}_2) \leftarrow \text{rng.UniformFloat}()$
18 $(i, j) \leftarrow (\lfloor N\mathcal{U}_1 \rfloor, \lfloor N\mathcal{U}_1 \rfloor)$   // Get two marginal distributions
19 $P^{22}_{\theta\alpha}(\tilde{h}, l, s_0) \leftarrow P_i(x_{\tilde{h}}, l_{\text{dist}}) P_j(y_{\tilde{h}}, l_{\text{dist}}) \frac{\alpha^2_{\text{dict}}}{\alpha_x\alpha_y} C$        // Eq. 11
```