Real-time Rendering of Procedural Multiscale Materials

Tobias Zirr*
NVIDIA
Karlsruhe Institute of Technology

Anton S. Kaplanyan†
NVIDIA

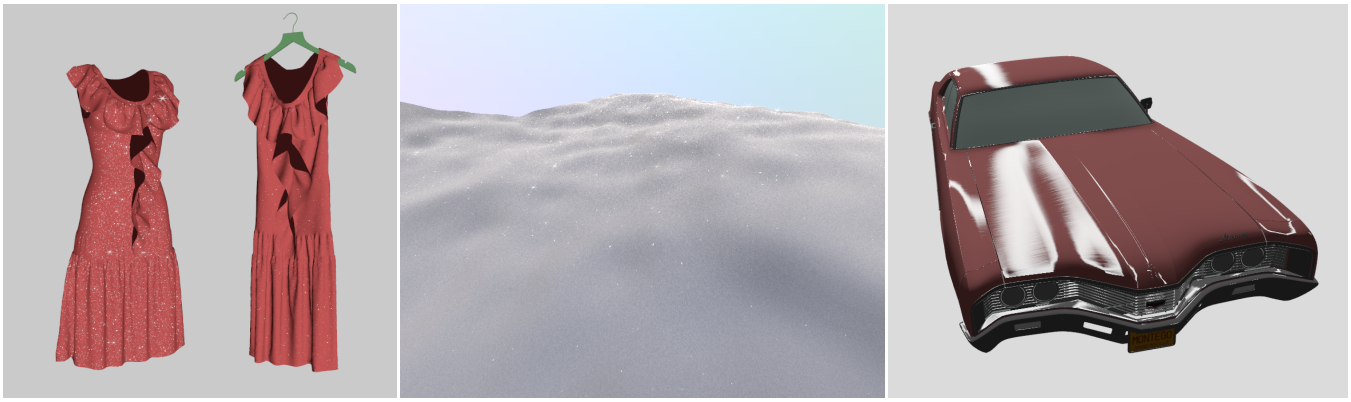

Figure 1: *Example of various materials with microdetails rendered with our procedural real-time approach on NVIDIA GeForce 980 GTX GPU. Left to right: (a) sparkling fabric on evening dresses (6.6ms/frame); (b) procedural terrain with grainy snow on an overcast day (7.8ms/frame); (c) brushed aluminum on a car (7.3ms/frame).*

Abstract

We present a stable shading method and a procedural shading model that enables real-time rendering of sub-pixel glints and anisotropic microdetails resulting from irregular microscopic surface structure to simulate a rich spectrum of appearances ranging from sparkling to brushed materials. We introduce a biscale Normal Distribution Function (NDF) for microdetails to provide a convenient artistic control over both the global appearance as well as over the appearance of the individual microdetail shapes, while efficiently generating procedural details. Our stable rendering approach simulates a hierarchy of scales and accurately estimates pixel footprint at multiple levels of detail to achieve good temporal stability and antialiasing, making it feasible for real-time rendering applications.

Keywords: Shading Models, Procedural Textures, Level of Detail

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

1 Motivation

Accurate rendering of photorealistic materials in real-time applications is a demanding topic. Physically based shading requires both the accuracy of underlying scattering models as well as the strictly constrained computational cost usually limited by a few milliseconds per frame on a modern Graphics Processing Unit (GPU).

Procedural discrete materials, such as glints, snow, and sand, have recently gained attention in both real-time rendering as well as in the offline rendering.

The state-of-the-art real-time methods [Bowles and Wang 2015; Michels et al. 2015] are lightweight and usually model simple sparkling appearance as an additional subtle trait of a material, making their appearance limited to sparse glints. Artist-made textures with conventional physically based shading supplement such approaches to simulate more complex effects, such as anisotropic highlights, grooves and other details, thus both sacrificing the multiscale appearance and increasing aliasing.

On the other hand, offline approaches [Jakob et al. 2014; Yan et al. 2014; Meng et al. 2015] provide superior quality and a wide gamut of appearances varying from anisotropic microgrooves caused by brushed materials to dense temporally-stable discrete glints that can transition into a smooth highlight at distance. While providing high quality appearance and advanced control over the material structure, such methods are not suitable for real-time applications.

In addition, none of the existing approaches allow the user to control the appearance of both microdetails of the local structure, as well as the distant appearance at large scale induced by these microdetails, such as shape and anisotropy of distant highlights.

In this work, we provide a spatially-varying multiscale appearance model with following properties:

- fully procedural and real-time evaluation of microdetails;
- anti-aliased and temporally stable shading using multilevel object-space grids;
- controlled appearance of both microdetails and distant material parameters;

*e-mail:tobias.zirr@kit.edu
†e-mail:akaplanyan@nvidia.com

ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games 2016, February 27-28, 2016, Redmond, WA
ISBN: 978-1-4503-ABCD-E/16/07
DOI: http://doi.acm.org/10.1145/9999997.9999999

- suitable for a wide range of materials, including glints, snow, and brushed metal.

We first overview the existing body of work on microfacet materials in Sect. 2, including multiscale materials and material filtering techniques, followed by an introduction of a biscale microfacet model we will employ for controlling the structure of our materials in Sect. 3. Next, we provide an efficient real-time evaluation using an approximation with multilevel grids in texture space and expected number of reflecting microdetails in Sect. 4. We evaluate performance and quality as well as discuss the limitations of the proposed approach in Sect. 5.

2 Previous Work

There is a large body of work on studying the light-matter interaction in both physics and computer graphics. We will cover here only the relevant computer graphics work that model light scattering on the surfaces. Both reflection and transmission of light through the surface are described by a Bidirectional Scattering Distribution Function (BSDF) f_s, which tells what part of irradiance $\mathrm{d}E$ from incident direction $\mathbf{i}$ at surface point is scattered into a particular outgoing direction $\mathbf{o}$ as

$$f_s(\mathbf{i},\mathbf{o}) = \frac{\mathrm{d}L(\mathbf{o})}{\mathrm{d}E(\mathbf{i})} = \frac{\mathrm{d}L(\mathbf{o})}{L(\mathbf{i})\,\mathrm{d}\mathbf{i}^{\perp}},$$

where $L(\mathbf{o})$ is the outgoing radiance and $L(\mathbf{i})$ is the incident radiance. This function describes local scattering at a surface point (the implicit dependence on surface point $\mathbf{x}$ is omitted hereafter) and therefore is a description of a material appearance commonly used in rendering.

Among many existing BSDF models there is an important class of microfacet surface scattering models. Torrance and Sparrow [1967] introduced a microfacet reflection model, which was adapted to graphics by Blinn [1977] as well as in a more generic form by Cook and Torrance [1982]. This model is based on the assumption that the scattering is caused by an underlying microsurface structure, which can be usually represented by a height field of microsurface heights and defined as a product of the following terms

$$f_s(\mathbf{i},\mathbf{o}) = \frac{F(\mathbf{i},\mathbf{o})D(\mathbf{m})G(\mathbf{i},\mathbf{o},\mathbf{m})}{4\,|\mathbf{i}\cdot\mathbf{n}|\,|\mathbf{o}\cdot\mathbf{n}|}, \tag{1}$$

where $|\cdot|$ denotes an absolute value of a dot product; $\mathbf{n}$ is the surface normal; $\mathbf{m}$ is a half-way vector between $\mathbf{i}$ and $\mathbf{o}$ that also represents the direction of a microfacet slope, D is a distribution of micronormals, called a *Normal Distribution Function (NDF)*, G is a masking-shadowing term, and F is a Fresnel term. The masking-shadowing term G estimates how many microfacets are visible from the given incident and outgoing directions $\mathbf{i}$ and $\mathbf{o}$.

Normal distribution function D at surface point $\mathbf{x}$ is defined as a probability of having a slope direction $\mathbf{m}$ on a field of slopes defined on a unit patch A as

$$D(\mathbf{m}) = \frac{\mathrm{d}A(\mathbf{m})}{A\,\mathrm{d}\mathbf{m}}, \tag{2}$$

where $A \equiv 1$ is the area of unit patch (by convention) and $\mathrm{d}A(\mathbf{m})$ is the area of all slopes with direction $\mathbf{m}$. We will also use an alternative notation where the NDF $D(\mathbf{x})$ is defined on the domain of slopes with $\mathbf{x} = \mathbf{m}_{xy}/\mathbf{m}_z$ being a projection of $\mathbf{m}$ onto a parallel plane one unit away from the surface along the normal (see Fig. 2 bottom). Even though this change of domain requires a corresponding Jacobian, we usually omit it in equations until later point to keep the notation clear. We will devote the rest of the section to the existing work on NDF, which is mainly responsible for the final appearance of the material. Beckmann NDF [1963] assumes to have a height field of microsurfaces with heights distributed according to a bivariate Gaussian distribution. Closed-form approximation for shadowing-masking term was also developed for this distribution [Smith 1967].

Trowbridge and Reitz [1975] proposed another distribution of slopes, also known in graphics as GGX [Walter et al. 2007], which corresponds to the distribution of slopes on an ellipsoid and provides a closer match for some measured materials [Burley 2012].

Materials with anisotropic highlights were first studied by Kajiya and Kay [1989], followed by works by Poulin and Fournier [1990] and Ward [1992]. A degenerate one-dimensional distribution of slopes on a microcylinder was studied in the context of hair rendering [Marschner et al. 2003]. Microfacet distributions as well as the masking term G were also extended for anisotropic materials, which corresponds to anisotropic scaling of the height field [Heitz 2014].

Filtering NDFs. The NDF of an advanced material, such as normal-mapped or displacement-mapped geometry with complex appearance, can be arbitrarily complex requiring careful evaluation to preserve the appearance and at the same time avoiding aliasing. The challenges of stable filtering as well as of preserving the appearance when rendering such NDFs were well recognized by the researchers. Representation of complex materials with multiple scales was studied by Westin et al. [1992]. Fournier [1992] proposed combining NDF produced by multiple surfaces and bump maps with the NDF of a surface BSDF. A smooth transition between displacement maps, bump maps, and simple BSDF shading at multiple scales was proposed [Becker and Max 1993] by using a hierarchy of multiple BSDF frequency levels as well as a modification to bump mapping. Kautz and Seidel [2000] proposed a shift-variant BSDF model to parameterize and precompute well-known BSDF models into a non-linear basis in order to efficiently filter and accelerate shading with such materials on a GPU. Toksvig [2005] proposed a practical way of computing a variation of normals in a prefiltered mip of a normal map texture by using a Gaussian NDF fit. More advanced filtering of normal maps includes storing multiple parametric NDF lobes motivated by frequency domain normal map filtering [Han et al. 2007], using a mixture of BSDFs [Tan et al. 2008], as well as linearly filterable moments of parametric distributions stored in mip chains of textures [Olano and Baker 2010]. A similar approach was also applied to gradually convert displacement mapping into an NDF at a distance [Dupuy et al. 2013]. A survey [Bruneton and Neyret 2012] discusses more advanced prefiltering methods. A related recent work [Nagano et al. 2015] proposes to simulate skin wrinkle deformation by scaling the displacement maps using convolution with anisotropic kernels.

Multiscale and Spatially-Varying NDFs. Recent advances in image quality allowed researchers to look into more sophisticated microfacet models with spatial variation, which can handle spatially as well as angularly varying NDFs with distinctly observable microstructure details, such as glints, grooves, and scratches at various scales. Bosch [2007] in his thesis provided multiple ad-hoc methods for measuring and rendering of vector-based scratches and grooves by checking the configurations of the groove geometry under pixel footprint, as well as derived closed-form approximations for reflectance and shadowing caused by grooves and scratches. Wu et al. [2011] proposed an approach for modeling materials using biscale material design, where a large-scale (macroscale in our work) appearance is derived from small-scale (microscale in our work) details designed by the user. In a follow-up work, the same

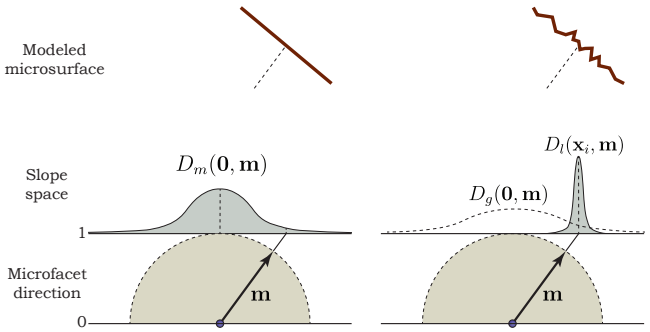

Figure 2: *Modeling biscale NDF in the domain of slopes (parallel planes). Left: first we draw a center $\mathbf{x}_i$ of a microdetail according to the density of microdetails D_m. Right: next we draw the actual slope around the selected center $\mathbf{x}_i$ according to the local roughness of a single microdetail D_l. Vector $\mathbf{m}$ is a microfacet (slope) direction on the hemisphere of surface tangent frame.*

authors [2013] proposed to reconstruct the small scale details out of a target macroscale appearance. Jakob et al. [2014] proposed a discrete spatially varying mesoscale NDF that stochastically models the number of discrete slopes covered by the pixel footprint and can be quickly evaluated by a hierarchical subdivision of the latter in microfacet domain. This method focuses on NDFs of discrete nature, thus is not suitable for anisotropic or elongated microdetails, such as grooves and scratches. Yan et al. [2014] concurrently proposed another method for handling spatially varying microstructure defined by a high resolution normal map texture. Their approach resolves a mesoscale NDF defined on a pixel footprint by hierarchically pruning irrelevant normal map texels. As we will show in Sect. 3, microdetails potentially affect the macroscale appearance, which might be undesirable for artistic control. Both methods are designed for high-quality offline rendering and are too computationally expensive for real-time applications.

We propose an optimized approximate method suitable for real-time applications that is completely procedural, provides artist-friendly control over appearance at all scales, and models a wide variety of materials, such as glints, granular surfaces, and anisotropic brushed grooves (Fig. 1).

3 Biscale Microfacet Model

First, similarly to Walter et al. [2007], Wu et al. [2011], Jakob et al. [2014], and Yan et al. [2014], we redefine an NDF in Eq. 2 by extending the unit patch A to define the NDF on an arbitrary patch, which in our case is a cell on the user-defined object-space[1] regular grid. This allows us to use a generalized multiscale notion of NDF, which tells us the probability density of a particular slope on this patch. This NDF can then be used directly for multiscale rendering with estimated primary footprint of a pixel [Igehy 1999] or a beam after multiple interactions [Bagher et al. 2012].

We first model a biscale NDF as a building block for a more general multiscale NDF that is composed of local biscale NDFs for every pair of adjacent levels. This will allow us to have a virtually infinite zoom into different scales of the slope field. We first model only two scales in the domain of slopes (also called a parallel plane domain, Fig. 2, left bottom) as follows. First the position $\mathbf{x}_i$ of a microscale detail center on a parallel plane is drawn from a *mesoscale* distribution of microdetails D_m, then the actual slope is placed around this center by drawing a value $\mathbf{y}_i$ from a *local* normals distribution for a single microdetail D_l, leading to the final slope. Since the final slope position is a sum of two random variables $\mathbf{x}_i$ and $\mathbf{y}_i$, their resulting density D_g is a convolution of two NDFs at different scales as

$$D_g(\mathbf{x}) = \int_{\mathcal{R}^2} D_m(\mathbf{y}) D_l(\mathbf{x}-\mathbf{y})\,\mathrm{d}\mathbf{y} = \langle D_m * D_l \rangle, \qquad (3)$$

where D_m controls the density of microdetails, whereas D_l controls the local appearance of a single microdetail. Similar models have been introduced before in a related context of normal map filtering [Han et al. 2007; Fournier 1992]. Multiple scales can be defined recursively by analogy.

With this definition, the support patch of the NDF has finite extent of one grid cell in order to handle locally correlated microdetails. Thus, there is a direct correspondence between the slope and its location on the support patch. However, later we will define a nested hierarchy of such grids, so that the actual location of a particular slope within the patch is procedurally generated on the fly.

Closed Form Convolutions for Existing NDFs. In order to efficiently evaluate all distributions in real time, it is preferable to have a closed form for convolution of two NDFs. The convolution in Eq. 3 can be computed in closed form for Beckmann distributions as a convolution of two bivariate Gaussian distributions in the parallel plane domain (Fig. 2), so given the relation between Beckmann roughness α and Gaussian standard deviation σ being $\alpha = \sqrt{2}\sigma$, the global roughness for distribution D_g can be computed as

$$\alpha_g^2 = \alpha_m^2 + \alpha_l^2, \qquad (4)$$

where α_g is a roughness of a resulting global Beckmann distribution D_g that gives the appearance at distance; α_l is the local roughness of a single individual microdetail; and α_m is the roughness of the mesoscale Beckmann distribution D_m of microdetails (see Fig. 2). All normalization equations (see, e.g., [Walter et al. 2007] or [Heitz 2014]) for energy conservation and physical plausibility hold as usual, because the resulting distribution is another Beckmann distribution.

Unfortunately, Trowbridge-Reitz distribution [Trowbridge and Reitz 1975] does not have a closed form convolution neither with itself nor with Beckmann distribution. Due to this reason, hereafter, we consider only Beckmann distributions for all working distributions throughout the paper because of their convenient convolution properties. In order to compute the final value of BSDF, we are using the Eq. 1 with appropriate Fresnel term and a Beckmann shadowing-masking term.

Decomposing Anisotropy. All terms in Eq. 4 are generally 2×2 covariance matrices. However, we consider only diagonal matrices, which is sufficient for representing arbitrary anisotropy in an orthonormal tangent frame up to a rotation. In this case, Eq. 4 also holds component-wise, along both $\mathbf{u}$ and $\mathbf{v}$ axes of the tangent plane. This allows us to control global anisotropic roughness independently from the local anisotropic roughness of a single microdetail separately for each axis. This can be done by selecting an appropriate per-axis roughness for mesoscale distribution D_m accordingly, which we will use in Sect. 4.4. Note that the resulting density D_g defines the global appearance of the material in the limiting case, when individual microdetails disappear at distance. This appearance is a desirable parameter of material for user to control, and thus it is exposed in our model.

[1] By object space we consider any reasonable two-dimensional parameterization of a surface that provides a deterministic, and, in case of grooves, piecewise continuous mapping from a surface point to a bounded two-dimensional domain. Texture mapping is one common choice.

One-Dimensional (Semi-Discrete) NDF. In the degenerate case when $\alpha_l \to \mathbf{0}$, each microdetail can take only the slope value at its center (around $\mathbf{0}$ in the local frame of a microdetail) turning the local NDF into a Dirac delta distribution

$$D_l(\mathbf{m}) = \delta_{\mathbf{m}}(\mathbf{0}). \tag{5}$$

This case represents discrete microdetails, such as glints [Jakob et al. 2014]. We extend this definition to model anisotropic slopes that are constant along one direction on the parallel plane, such as long grooves on brushed materials. The Dirac delta in Eq. 5 denotes a deterministic selection of a microfacet slope at the center. Since we model our NDF in the domain of slopes, we can reparameterize and expand the Dirac delta in this domain along $\mathbf{u}$ and $\mathbf{v}$ axes as

$$\delta_{\mathbf{m}}(\mathbf{m}_0) = \delta^{\parallel}(\mathbf{x}_0) \left|\frac{\mathrm{d}\mathbf{m}}{\mathrm{d}\mathbf{x}}\right| = \delta_{\mathbf{u}}(u_0)\delta_{\mathbf{v}}(v_0) \left|\frac{\mathrm{d}\mathbf{m}}{\mathrm{d}\mathbf{x}}\right|,$$

where $\delta^{\parallel}$ is the Dirac delta in the domain of slopes (parallel plane domain), and the Jacobian transformation from the microfacet direction to the domain of slopes is

$$\left|\frac{\mathrm{d}\mathbf{m}}{\mathrm{d}\mathbf{x}}\right| = |\mathbf{m}\cdot\mathbf{n}|^4,$$

where $\mathbf{n}$ is the surface normal; and $\mathbf{m}$ is the unit vector of microfacet direction. Here, $\delta_{\mathbf{u}}(u_0)$ and $\delta_{\mathbf{v}}(v_0)$ are the canonical one-dimensional Dirac deltas on the axes of a tangent plane at u_0 and v_0 respectively. They represent a deterministic choice of the x_u and x_v components of the slope $\mathbf{x}$ on the parallel plane. Now we can extend a two-dimensional discrete distribution D_l to a line (one-dimensional discrete) distribution D_l^1 on this domain by mollifying one of the Dirac deltas into a one-dimensional Gaussian distribution (for example, along $\mathbf{v}$) as

$$D_l^1(\mathbf{x}) = \delta_{\mathbf{u}}(u_0)\mathcal{B}^1_{\mathbf{v}}(v_0, \alpha_{\mathbf{v}})\,|\mathbf{m}\cdot\mathbf{n}|^4, \tag{6}$$

where $\mathcal{B}^1_{\mathbf{v}}$ is a one-dimensional analog of Beckmann distribution with scalar roughness $\alpha_{\mathbf{v}}$ along $\mathbf{v}$ axis, which can be written as a one-dimensional Gaussian distribution $\mathcal{B}^1_{\mathbf{v}}(v_0, \alpha_{\mathbf{v}}) = \mathcal{N}_{\mathbf{v}}(v_0, \alpha_{\mathbf{v}}/\sqrt{2})$. One-dimensional distribution along $\mathbf{u}$ is obtained analogously. Note that the normalization factor of $\mathcal{B}^1$ is now $1/\sqrt{\pi}$ instead of $1/\pi$ as in the conventional two-dimensional Beckmann. This NDF models long grooves, where the slope stays constant along one axis, while along another axis it has Beckmann-distributed variation. Even though, in order to faithfully represent fixed-location grooves, we would need to apply a spatial Dirac delta to lock them to particular surface positions [Jakob et al. 2014], in practice, we will always select the scale such that the correlation between location and slope can be neglected. We will use this formulation of one-dimensional NDF to simulate highly anisotropic effects such as brushed grooves. To our knowledge, this is the first time the semi-degenerate one-dimensional NDF was defined for linear distributions of slopes.

From Microscale to Macroscale. Recall that by design our NDF in Eq. 3 is defined on a finite-sized cell of a regular object-space grid with user-controlled density. This induces a natural transition from microscale to macroscale. If a microdetail gets larger than the considered rendering region, such as the pixel footprint, the microdetail is supposed to span across multiple regions. This means that after some scale, individual microdetails become observable. In the limiting case of anisotropic one-dimensional distribution, the anisotropy spans across the object, i.e., the selection of a groove is done only once along the degenerate axis to guarantee the continuity of a groove. In Sect. 4.4 we will make sure that the large microdetails are observed as continuous grooves at macroscale by extending the patch for such microdetails to be long enough to accommodate the extents of the microdetails.

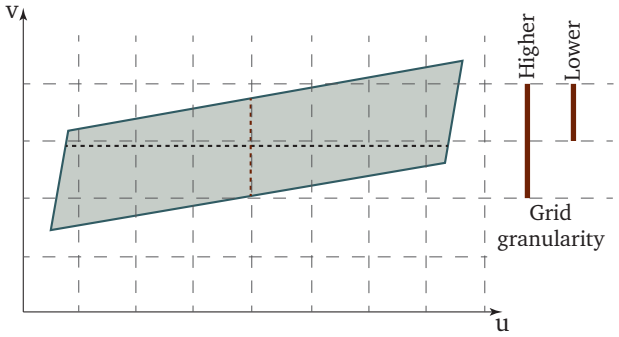

Figure 3: *Pixel footprint is approximated in texture space to a first order using ray differentials as a parallelogram (blue). High and low grid cell sizes are selected based on the minor extent of the parallelogram by rounding to the closest power of two from above and from below (red lines).*

In the next section we will discuss how to practically define a multilevel grid for infinitely many microdetails using a biscale NDF as a building block, and how to evaluate NDF defined on cells within each grid level.

4 Real-time Shading with Microdetails

First, we consider a regular grid in object space parameterization; practically we rely on an existing uv texture mapping. The grid is defined with user-specified density, which is a global scalar that specifies the density of microdetails per unit area in uv space, thus setting the granularity of the grid, i.e., its cell size. We use this grid level as a reference level.

In order to simulate multiscale NDF, we consider a hierarchy of such grids, where each next grid level is twice as dense in every dimension as the previous one, similarly to texture mip chain. We employ the procedural nature of our model: the microdetails are independently and identically distributed. This means that the appearance should have the same distribution, but it is not required to refine exactly the same realization from this distribution at multiple scales, as long as we preserve the overall statistics of the distribution at all scales. In Section 4.3 we evaluate the expected number of microdetails such that the distribution is preserved across all levels. Thus, we use a biscale NDF at every grid level and define a nested multilevel NDF with infinite number of scales this way.

We evaluate this multilevel NDF by estimating the total area of finite-sized microdetails covered by each pixel (*pixel footprint*) that are reflecting the light. There are two challenging aspects for the real-time evaluation of the biscale NDF on each grid: (1) pixel footprint can potentially cover an arbitrarily large area in the multiscale hierarchy; and (2) the search is four-dimensional: first we need to collect all covered cells on the surface, then we need to compute the area of actively reflecting microfacets.

In order to tackle the first problem, we quantize the pixel footprint on each grid and then estimate the reflecting area. For every pixel, we consider two adjacent texture-aligned grid levels with scales based on the pixel footprint (see Fig. 3). Similarly to the mip level selection, one grid is the closest coarser level and one is the closest finer level for the projected pixel granularity. Then, similarly to anisotropic texture filtering, we evaluate shading at all covered

```
pixel P
H = normalize(L + V), Hp = paraboloid_map(H)
// Normalization for a single microdetail (Eq.7)
m0 = Dl(0)
// Probability mass of reflecting facets (Eq.7)
pmf = Dg(H) / m0
find grids G[0], G[1] in footprint(P)
reflectance = 0
for i = 0, 1:
 foreach cell C in G[i] intersecting P:
  // Quantize perturbed paraboloid facet orientation
  Hidx = quantize(Hp + m0 * rand(C))
  // Number of microdetails inside cell
  Cn = area(C) * density
  // Randomize scale of microdetails
  Cn = Cn * abs(gauss(rand(Hidx + C), 1, mdsVariance))
  // Draw number of reflecting microdetails
  rn = binomial(rand(Hidx + C), Cn, pmf)
  // Cell reflectance
  Cr = rn * m0 / Cn
  // Compute bilinear cell weight
  Cw = approximateCoverage(C, P)
  // Blend between grid levels
  reflectance += mipWeight(G[i], P) * Cw * Cr
```

Listing 1: *High-level pseudocode for simulating microdetail by counting within each grid cell covered by a pixel. Note that* `rand(..)` *is a deterministic quasi-random number generator that provides the same results for the same seed.*

cells in both grids and blend between the shading results in each cell. This ensures stable anti-aliased glints at arbitrary scales.

Our multilevel grid approach, while being approximate, provides a significantly more efficient evaluation compared to the hierarchical search processes used by Jakob et al. [2014] and Yan et al. [2014]. While their hierarchical approaches lead to perfectly coherent results that guarantee area preservation across all scales, they greatly increase the complexity of the search.

In order to tackle the second problem, the problem of finding reflecting microfacets in the four-dimensional space, (within pixel footprint and within parallel plane given the reflection direction) we first split it into two steps: (1) estimate the number of microfacets within a cell; (2) estimate the expected number of reflecting microfacets out of these. This way, we partition the total number of microdetails into reflecting and non-reflecting microdetails inside each cell. Instead of handling each microdetail separately, we compute the expected number of reflecting microdetails within the region, and then draw the final number of reflecting microfacets using stable random process. The cone of reflecting microdetail orientations is determined by the roughness of the microdetails and the size of the light source (see Sect. 4.3). The final partitioning for a particular cell is done using a binomial random process $B(\xi, n, P)$. The selection probability P corresponds to the probability mass of a continuous Beckmann distribution of microdetails inside the reflection cone. The process is seeded with the distinct noise value ξ of each cell, giving deterministic and thus stable results. Listing 1 provides pseudocode for the main algorithm.

4.1 Coherent Multiscale Grids

We use an interleaved hashing scheme of cell indices that ensure that the same seeds occur at similar locations across all scales. This ensures the blending between the results of the random processes at different scales always retains parts of the features on both levels.

Since all our grid scales are powers of two, our hashing scheme needs to ensure that the indices of a coarser grid reappear with double spacing in the finer grid. We achieve it by making the result of

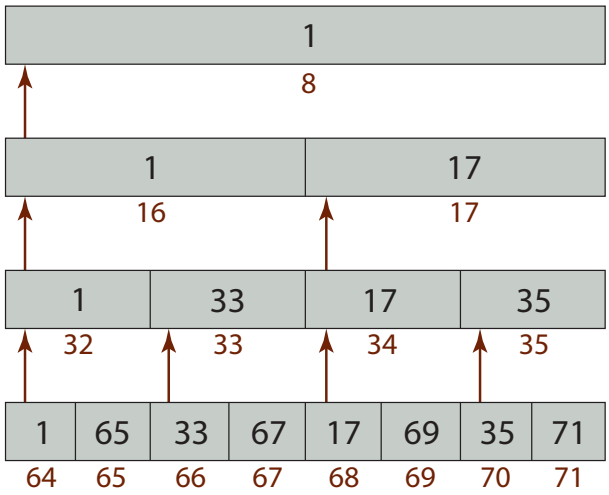

Figure 4: *Nested power-of-two grids with mapping from per-grid cell indices (red) to coherent indices (given inside the cells). Note how one out of two generated indices is always preserved between coarser and finer levels across all scales.*

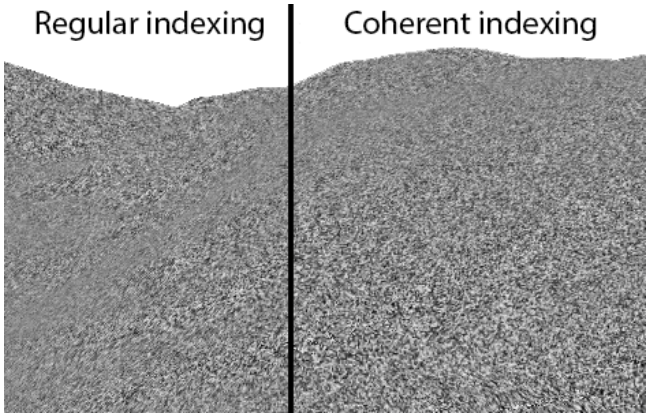

Figure 5: *Regular noise with naïve non-coherent blended noise grids (left) and the proposed large-scale coherent noise grid (right).*

the hashing invariant under input multiplication by two. Instead of multiplying the grid ids at each level multiple times (also, we do not have a root level), we reverse the doubling process and bit-shift the index of each grid cell to the right until the least significant bit is a one (`i >> trailingZeros(i)`). This ensures we have a coherent noise throughout all levels (see Fig. 4 for illustration). While this leads to some recurring patterns for small grid indices, we decorrelate the results by adding an arbitrary constant coordinate offset at every level.

4.2 Filtering with Anisotropic Pixel Footprint

We approximate the pixel footprint in texture space by a parallelogram formed by the screen-space derivatives of the texture coordinates uv (see Fig. 6). We compute the cell size s_c using the shorter of the distances between the parallel edges projected to each texture axis using the max-norm of the inverted derivative matrix:

$$s_c = \left(\| (M_{xy}\ uv)^{-1} \|_{\max} \right)^{-1}.$$

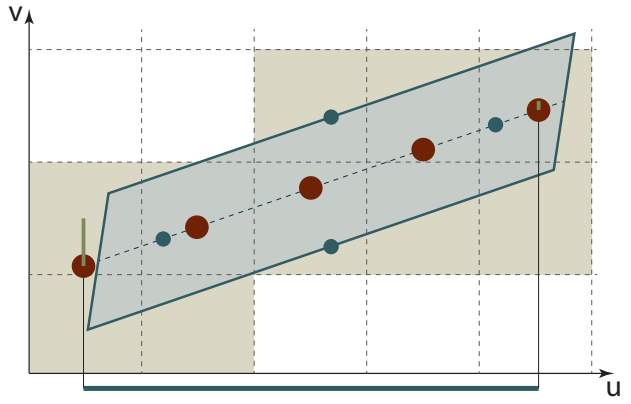

Figure 6: *A pixel footprint's parallelogram is defined by its edge centers (blue circles). The two side centers are shifted by half a cell inwards along the diagonal to provide a tight range of grid cells. We sample along the longer median of the parallelogram. On the median, the samples are snapped to the cell centers of our discrete grid along the longer extent. Number of samples is determined in the spirit of anisotropic texture filtering along the major extent with the steps that are defined by the minor extent. The olive thick lines on the outside shows how far the samples are from the cell centers. This distance is used for bilinear weights.*

Afterwards, we round the cell size to the next power of 2. In each of the corresponding grids, we step along the longer median of the pixel parallelogram, accumulating the results of all cells along the way (see Fig. 6). We use the bilinear interpolation weights corresponding to the offsets of the steps from the cell centers as an approximate value for the pixel coverage of each cell. The coverage is then used as a weight for the contribution sampled in each cell. We normalize by the total weight afterwards to handle multiple anisotropic line samples. In order to properly blend along the longer axis, the first and last steps are offset towards the pixel center in texture space by half a cell size each.

In order to avoid excessive tapping, we limit the anisotropy to a factor of K_a by enforcing $\|M_{xy}\ uv\|_{\max}/K_a$ as a lower bound for the cell size c_s, which corresponds to a fraction of the axis-aligned box that bounds the pixel's edge centers in texture space. In practice we use a maximum allowed anisotropy of $K_a = 16$ in all our experiments, which appears to be sufficient. This is analogous to the anisotropy clamping in texture filtering to limit the number of tapped texels. Our motivation is the same, and, notably, the maximum practical anisotropy level is also similar to the maximum level commonly used in anisotropic texture filtering.

4.3 Sampling the Number of Reflecting Microdetails

Half Vector Quantization. We project the half vector at the current pixel center to a plane using a paraboloid mapping. This keeps the density of half vectors approximately uniform in solid angle. Afterwards, we discretize this plane into multiple cells, the index of which can then be used for microdetail orientation-dependent seeding. This is an efficient way to approximate a spherical grid of half vectors. We choose the slope-domain cell size proportional to the sum of microroughness and light source cone angle (see Fig. 8) to approximate the complex integration over all slope-domain grid cells that contribute due to either high microroughness or larger light cones.

Now that we have a cell in object space and a current cell in half vector space, we need to estimate the expected number of reflecting microdetails within these current cells. We first compute the probability whether a microdetail reflects light by stochastically estimating the number of such microdetails based on their density in parallel plane domain. However, we need to extend it to handle non-specular microdetails. Since we are working with rough microdetails, all microdetails reflect at least a fraction of the light. Therefore, we cannot make a deterministic decision whether an individual microdetail contributes or not. Instead, for a single microdetail $\mathbf{m}$, we turn the continuous decline of contribution into a probability of the microdetail centered at $\mathbf{m}$ reflecting for half vector $\mathbf{h}$ as

$$P_{\mathbf{m}}(\mathbf{h}) = \frac{D_l(\mathbf{h}-\mathbf{m})}{D_l(\mathbf{0})}.$$

Here, we make an assumption that the local density of microdetails within the current half-vector cell is relatively flat. Thus, we can compute the overall probability mass $P(\mathbf{h})$ of contributing microdetails as

$$\begin{aligned} P(\mathbf{h}) &= \int_{\mathcal{R}^2} D_m(\mathbf{m}) P_{\mathbf{m}}(\mathbf{h})\,\mathrm{d}\mathbf{m} \\ &= \int_{\mathcal{R}^2} D_m(\mathbf{m}) \frac{D_l(\mathbf{h}-\mathbf{m})}{D_l(\mathbf{0})}\,\mathrm{d}\mathbf{m} = \frac{D_g(\mathbf{h})}{D_l(\mathbf{0})}, \end{aligned} \tag{7}$$

here, as an exception, we use $\mathbf{m}$ and $\mathbf{h}$ as points on the parallel plane of slopes for clarity of convolution (in the rest of the work they are used as unit vectors). We use the resulting probability mass as the second parameter of the binomial distribution. We cancel out $D_l(\mathbf{0})$ after the binomial sampling. Note in the limit this results in the correct global BSDF.

So far, apart from some variation due to the global NDF D_g, our random process gives the same results for all microdetail orientations. This is unrealistic, since a surplus in microdetails of one orientation has to be compensated by a shortage of microdetails with different orientations in order to conserve the overall surface area. While we do not enforce area conservation, we apply an additional randomization seeded by the microdetail orientation. Thus, the expected total area is constant and stays conserved on average.

Sampling the Binomial Distribution. In order to be able to quickly draw the number of reflecting microdetails N from the binomial distribution at line 18 in Listing 1, we only compute the exact distribution for small expected numbers. Beyond that, we switch to a Gaussian distribution using the first and second moments of the binomial distribution for fitting. Additionally, we enforce an upper bound on the number of iterations in the binomial draw by replacing the tail with an exponential distribution after a fixed maximum number i of iterations:

$$N = i + 1 + \log_p\left(1 - C\frac{U - P(N \le i)}{1 - P(N \le i)}\right), \text{with}$$
$$C = 1 - p^{N_{\max}-i-1}.$$

Microdetail Size Variation. With a fixed microdetail density, interesting results are restricted to a rather small range of scales. In order to achieve rich appearance across a large range of scales (Fig. 7), e.g., from close-up across an entire landscape, we additionally randomize the size, and thus the density of microdetails in each grid cell (lines 13-16 in Listing 1). We use a Gaussian distribution to model the randomization of the number of microdetails in each cell.

4.4 Anisotropic Microdetails

In order to distribute anisotropic microdetails and preserve the global appearance as discussed in Sect. 3, we use Beckmann distributions. Given the user-defined global roughness α_g and the local

Figure 7: DRESS *scene rendered with a glittery material with the same density of microdetails with varying sizes of microdetails. Left: constant size of microdetails; right: user-tweaked Gaussian variation of microdetail size.*

roughness of a single microdetail α_l we can derive the mesoscale parameter α_m of the Beckmann distribution of microdetails based on the convolution in Eq. 4 as

$$\alpha_m = \sqrt{\alpha_g^2 - \alpha_l^2}.$$

It is important to note that in order for the global roughness α_g to be possible, the local roughness of a single microdetail is limited by $\alpha_l \leq \alpha_g$. In order to handle one-dimensional NDFs in Eq. 6, we compute the quantities directly in the parallel plane domain and then apply the transformation Jacobian.

We compute the anisotropy factor γ_m of the distribution of microdetails along each axis as the ratio of the roughnesses on the axes $\mathbf{u}$ and $\mathbf{v}$ as

$$\gamma_m = \alpha_m(\mathbf{u})/\alpha_m(\mathbf{v}).$$

This is equivalent to stretching the surface with isotropic roughness $\alpha_m(\mathbf{u})$ by γ_m along $\mathbf{v}$ [Heitz 2014]. In order to provide continuity for elongated anisotropic microdetails (observed in macroscale as discussed in the last paragraph of Sect. 3), we apply this stretching to the texture space grids along the corresponding axes after footprint estimation and grid level selection.

It is important not to apply NDF anisotropy before that, otherwise the anisotropic pixel footprint filtering will fight against the NDF anisotropy: While the former tries to make up for the mismatch of texture space grid and screen resolution by filtering, here we deliberately induce an anisotropic mismatch of cell sizes to allow for correlation across multiple pixels in the case of anisotropic microdetail distributions.

In order to obtain anisotropic microdetails, we stretch both the texture coordinates of the pixel center and the pixel footprint used for determining the covered grid cells. Care has to be taken in order not to unnecessarily increase the number of taps taken by the grid-based filtering. For that, we always downscale texture coordinates. Stretching one axis with a stretch factor greater than one is equivalent to stretching the other axis with the reciprocal factor.

4.5 User Parameters

In order to provide a convenient control over the appearance at multiple scales, we expose the following user parameters: the roughness of the macroscopic material, the roughness of a single microdetail, the density (number of microdetails per unit object-space area), the microdetail scale variation, and the overall intensity. Both roughness parameters can be anisotropic and exposed along u and v tangent plane's axes separately. Since the roughness of a single microdetail is bounded by the global roughness, we expose the microdetail roughness scaled relatively to the current global roughness values for convenience. This allows to easily steer anisotropic glints by setting the fractional roughness along one direction to 1. The density is specified on a logarithmic scale to allow for more linear control over the apparent density.

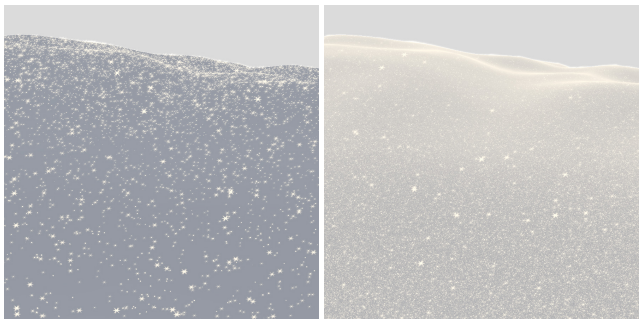

Figure 8: *Glittery snow material on a terrain with varying-sized light source. Light source cone size varies from a point light (left) to an almost overcast illumination (right).*

For colored glints or grooves, one can also pick an albedo color for a microdetail out of a user palette. We demonstrate a uniform chromatic distribution by drawing a random albedo from the color wheel in each cell. In order to comply with the variance caused by the uniform color distributions of N microdetails previously drawn from the binomial distribution, the coloring is normalized with a factor of $1/\sqrt{N}$.

5 Results and Discussion

In this section we evaluate the performance as well as visual results of the proposed approach. The real-time performance of the OpenGL 4.5 implementation of our method is evaluated on multiple scenes using an NVIDIA GeForce 980 GTX GPU and the timings are presented in Table 1. All timings are in milliseconds per frame and measured for 1920×1080 (1080p) image resolution with vertical synchronization switched off. It is generally difficult to measure the shading performance, as it varies depending on the screen coverage, amount of overdraw, and the amount of other shading computations in the shader. Therefore we first measured a simple plane (no overdraw) and covered full screen with it at different viewing angles. In order to measure the pure shading time, we first measure the frame time without our approach to account for bandwidth, synchronization and all the additional passes for frame rendering and take this measure as a baseline. Then we enable our approach and report the time difference. Variance in timing for different scenes

Scene	Polys	Steep angle	Grazing angle
SCREEN PLANE	2	0.9	2.9
SNOW	32k	2.5	4.0
DRESS	100k	1.4	4.4
CAR (GROOVES)	570k	2.5	3.9
CRYTEK SPONZA	262k	3.0	5.9

Table 1: *Performance in milliseconds for intra-frame shading overhead at 1080p HD resolution, measured on NVIDIA GeForce 980 GTX GPU for scenes with specified polygon count (second column). Top table shows measurements of multiple scenes with various viewing angles: steep (perpendicular view), grazing (about 10 degrees to the surface, maximum 16× anisotropy).*

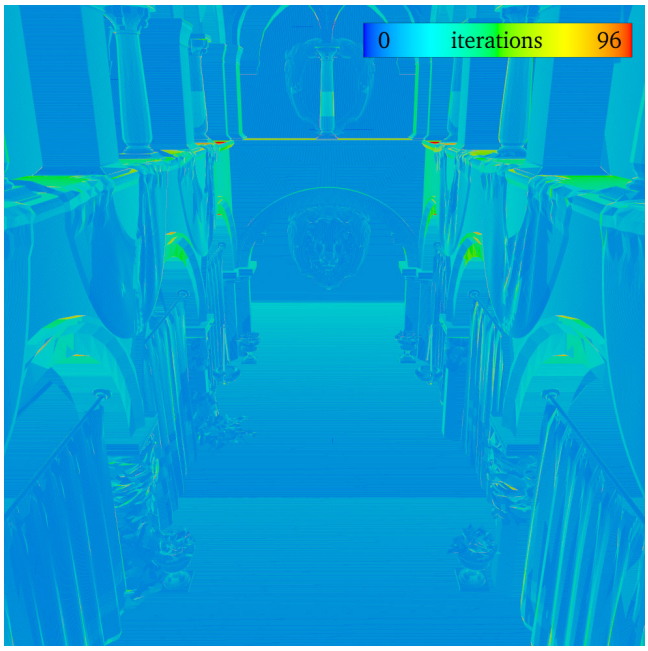

Figure 9: *False colored visualization of the number of cells visited by our algorithm (number of iterations) for varying pixel footprint in* CRYTEK SPONZA *scene.*

mostly comes from the amount of overdraw, thus our approach can potentially benefit from deferred shading, where the shading overdraw is almost non-present.

A false-colored visualization of number of iterations for the innermost `foreach` loop in Listing 1 is provided for CRYTEK SPONZA scene in Fig. 9. It demonstrates that our method requires more iterations to accurately resolve microdetails at grazing angles. Even though pixel footprint can theoretically cover arbitrarily many cells, in practice we limit the maximum anisotropy to the ratio of $K_a = 16$, thus limiting the maximum number of iterations. For the best-to-worst case scenarios of a plane observed at different angles (first row in Table 1) performance ranges by 2 milliseconds from the minimum of 8 cells to the maximum of 64 cells required by our method. The shader has 412 static instructions according to the `GLSLC` disassembling utility, 204 of which are inside the innermost `foreach` loop of Listing 1.

Fig. 10 demonstrates the ability of our approach to handle materials with dense microdetails, including granular materials, such as sand and snow, as well as glitter, and an anisotropic example of colored brushing. In all images we use a simple star-shaped posteffect to better convey the dynamic range of the sparkles.

Please see the supplementary video for the demonstration of temporal stability as well as the user control over the appearance. Note that some aliasing can be still observed in highly curved places, like the left front pillar of car's frame, which is caused by the macroscale NDF.

Our method is completely procedural, thus does not require any memory traffic, which makes it a perfect candidate for interleaving with the conventional real-time shading workload. Our approach is also very modular, thus is readily available for a plug-in integration into a shading system of an existing large-scale application or game engine.

Limitations and Future Work. We rely on parameterization of the mesh (texture uv mapping) for orientation and microdetail density. This requires a reasonable parameterization without singular or degenerate primitives. Stretched parameterization can cause unnecessary anisotropic filtering in texture space, thus hurting the performance, and, in case of anisotropic microdetails, such as brushed grooves, it can change the density of microdetails in such places.

Our work also relies on a common assumption about the pixel footprint and works with its first-order parallelogram approximation. The computation of the footprint can be further improved to provide more accurate results, e.g., by accounting for surface curvature that compresses the footprint. Anisotropy of pixel is limited by $K_a = 16\times$ of maximum aspect ratio, which suggests a potential future work in analytic integration of stable shading across the arbitrary footprint shape. We found that practically the current footprint computations are usually sufficient for our needs even in the cases of extreme anisotropy. Pixel footprint is also defined only for primary rays. The notion of an NDF patch can be extended to handle multiple indirect bounces (e.g., in the spirit of [Bagher et al. 2012]) as another potential future work.

Another limitation is that our method requires a practically computable convolution of two NDFs, which was the reason for considering only Beckmann NDFs throughout this work. Approximate numerical fit for convolution with a Trowbridge-Reitz distribution can be an important potential future work.

One interesting future work is to apply the quantized shading grids to conventional real-time shading to improve shading stability.

We also envision other applications for stable multiscale object-space noise, such as procedural placement of instances (e.g., grass, trees, etc.), and local procedural object-space content (e.g., various procedural texture maps, decals, and other object details).

6 Conclusion

In this paper, we propose a practical real-time method for stable antialiased rendering of challenging materials, such as glints and brushed grooves. Our biscale NDF model allows simulation and control of the material with microdetails by exposing separate controls for macroscopic appearance as well as for the appearance of local microdetails. We provide a stable multiscale noise generation pattern in object space, which is temporally stable and passes information across scales. We also use an accurate filtering across the anisotropic pixel footprint in object space to ensure robust sampling of microdetails at grazing angles of view. We demonstrate that our method can achieve a wide range of microdetail appearances, ranging from glitter and snow to brushed metal. In addition, we show that the method is readily applicable to modern real-time lighting models, such as light sources with finite angular extent. Our performance numbers indicate that the method is feasible to use in current and upcoming real-time applications.

7 Acknowledgments

We would like to thank Aaron Lefohn for supporting the research project within NVIDIA; Eric Enderton, Aaron Lefohn, Tim Foley, Chris Wyman, Anjul Patney, Jan Kautz, Marco Salvi, and Alexander Reshetov for the helpful discussions; Nir Benty for helping with the rendering infrastructure, Christoph Kubisch for the GLSLC tool, as well as the anonymous reviewers for their valuable feedback. Thanks to Anjul Patney for modeling the I3D scene, and Frank Meinl and Efgeni Bischoff for remodeling CRYTEK SPONZA. DRESS (Archmodels vol. 102) and CAR (17 x

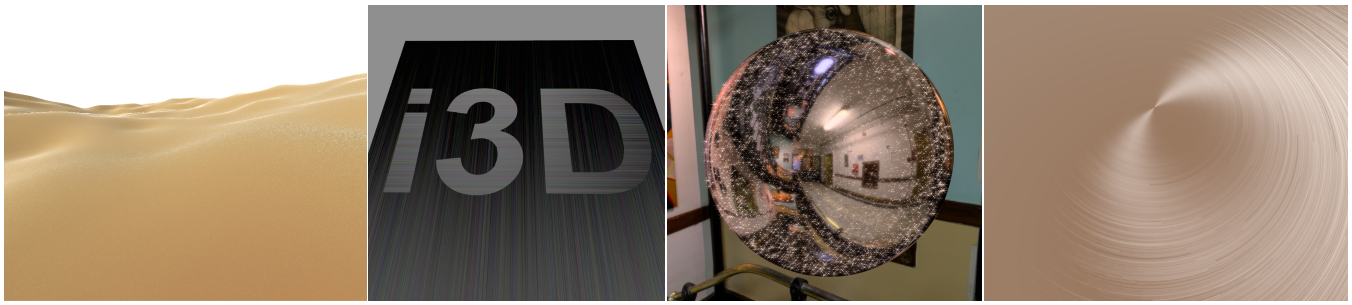

Figure 10: *Examples of high-density microdetails: (a) sand on a sunny day; (b) colorful anisotropic brushed grooves; (c) glitter illuminated by environment mapping; and (d) circular colored brushing.*

American Classic Cars) commercial models are provided by http://www.cgriver.com.

References

BAGHER, M. M., SOLER, C., SUBR, K., BELCOUR, L., AND HOLZSCHUCH, N. 2012. Interactive rendering of acquired materials on dynamic geometry using bandwidth prediction. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, 127–134.

BECKER, B. G., AND MAX, N. L. 1993. Smooth transitions between bump rendering algorithms. In *Proceedings of Computer graphics and interactive techniques*, Computer Graphics (Proc. SIGGRAPH), 183–190.

BECKMANN, P., AND SPIZZICHINO, A. 1963. *The scattering of electromagnetic waves from rough surfaces*. International series of monographs on electromagnetic waves. Pergamon Press.

BLINN, J. F. 1977. Models of light reflection for computer synthesized pictures. *Computer Graphics (Proc. SIGGRAPH) 11*, 2, 192–198.

BOSCH, C. 2007. *Realistic Image Synthesis of Surface Scratches and Grooves*. PhD thesis, Universitat Politcnica de Catalunya. 2007LIMO4021.

BOWLES, H., AND WANG, B. 2015. Sparkly but not too sparkly: A stable and robust procedural sparkle effect. In *Advances in Real Time Rendering, Part 1*, N. Tatarchuk, Ed. ACM SIGGRAPH Courses.

BRUNETON, E., AND NEYRET, F. 2012. A survey of nonlinear prefiltering methods for efficient and accurate surface shading. *IEEE Transactions on Visualization and Computer Graphics 18*, 2, 242–260.

BURLEY, B. 2012. Physically-based shading at Disney. *ACM SIGGRAPH Courses*, 10:1–10:7.

COOK, R. L., AND TORRANCE, K. E. 1982. A reflectance model for computer graphics. *Computer Graphics (Proc. SIGGRAPH) 1*, 1, 7–24.

DUPUY, J., HEITZ, E., IEHL, J.-C., POULIN, P., NEYRET, F., AND OSTROMOUKHOV, V. 2013. Linear efficient antialiased displacement and reflectance mapping. *ACM Transactions on Graphics (Proc. SIGGRAPH) 32*, 6, 211:1–211:11.

FOURNIER, A. 1992. Normal distribution functions and multiple surfaces. In *Graphics Interface Workshop on Local Illumination*, 45–52.

HAN, C., SUN, B., RAMAMOORTHI, R., AND GRINSPUN, E. 2007. Frequency domain normal map filtering. *ACM Transactions on Graphics (Proc. SIGGRAPH) 26*, 3.

HEITZ, E. 2014. Understanding the masking-shadowing function in microfacet-based BRDFs. *Journal of Computer Graphics Techniques (JCGT) 3*, 2, 48–107.

IGEHY, H. 1999. Tracing ray differentials. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH, 179–186.

JAKOB, W., HAŠAN, M., YAN, L.-Q., LAWRENCE, J., RAMAMOORTHI, R., AND MARSCHNER, S. 2014. Discrete stochastic microfacet models. *ACM Transactions on Graphics (Proc. SIGGRAPH) 33*, 4, 115:1–115:10.

KAJIYA, J. T., AND KAY, T. L. 1989. Rendering fur with three dimensional textures. *Computer Graphics (Proc. SIGGRAPH) 23*, 3, 271–280.

KAUTZ, J., AND SEIDEL, H.-P. 2000. Towards interactive bump mapping with anisotropic shift-variant BRDFs. In *Proc. SIGGRAPH/Eurographics Workshop on Graphics Hardware*, 51–58.

MARSCHNER, S. R., JENSEN, H. W., CAMMARANO, M., WORLEY, S., AND HANRAHAN, P. 2003. Light scattering from human hair fibers. In *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 22, 780–791.

MENG, J., PAPAS, M., HABEL, R., DACHSBACHER, C., MARSCHNER, S., GROSS, M., AND JAROSZ, W. 2015. Multiscale modeling and rendering of granular materials. *ACM Transactions on Graphics (Proc. SIGGRAPH) 34*, 4.

MICHELS, A. K., SIKACHEV, P., DELMONT, S., DOYON, U., MAHEUX, F., BUCCI, J.-N., AND GALLARDO, D. 2015. Labs R&D: Rendering techniques in Rise of the Tomb Raider. In *ACM SIGGRAPH 2015 Talks*, 81:1–81:1.

NAGANO, K., FYFFE, G., ALEXANDER, O., BARBIÇ, J., LI, H., GHOSH, A., AND DEBEVEC, P. 2015. Skin microstructure deformation with displacement map convolution. *ACM Transactions on Graphics 34*, 4, 109:1–109:10.

OLANO, M., AND BAKER, D. 2010. LEAN mapping. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, 181–188.

POULIN, P., AND FOURNIER, A. 1990. A model for anisotropic reflection. *Computer Graphics (Proc. SIGGRAPH) 24*, 4, 273–282.

SMITH, B. 1967. Geometrical shadowing of a random rough surface. *IEEE Transactions on Antennas and Propagation 15*, 5, 668–671.

TAN, P., LIN, S., QUAN, L., GUO, B., AND SHUM, H. 2008. Filtering and rendering of resolution-dependent reflectance models. *IEEE Transactions on Visualization and Computer Graphics 14*, 2, 412–425.

TOKSVIG, M. 2005. Mipmapping normal maps. *Journal of Graphics, GPU, and Game Tools 10*, 3, 65–71.

TORRANCE, K. E., AND SPARROW, E. M. 1967. Theory for off-specular reflection from roughened surfaces. *Journal of the Optical Society of America 57*, 9, 1105–1112.

TROWBRIDGE, T. S., AND REITZ, K. P. 1975. Average irregularity representation of a rough surface for ray reflection. *Journal of the Optical Society of America 65*, 5, 531–536.

WALTER, B., MARSCHNER, S., LI, H., AND TORRANCE, K. 2007. Microfacet models for refraction through rough surfaces. In *Proc. Eurographics Symposium on Rendering*, 195–206.

WARD, G. J. 1992. Measuring and modeling anisotropic reflection. *Computer Graphics (Proc. SIGGRAPH) 26*, 2, 265–272.

WESTIN, S. H., ARVO, J. R., AND TORRANCE, K. E. 1992. Predicting reflectance functions from complex surfaces. *Computer Graphics (Proc. SIGGRAPH) 26*, 2, 255–264.

WU, H., DORSEY, J., AND RUSHMEIER, H. 2011. Physically-based interactive bi-scale material design. *ACM Transactions on Graphics 30*, 145, 145:1–145:10.

WU, H., DORSEY, J., AND RUSHMEIER, H. 2013. Inverse bi-scale material design. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia) 32*, 6, 163:1–163:10.

YAN, L.-Q., HAŠAN, M., JAKOB, W., LAWRENCE, J., MARSCHNER, S., AND RAMAMOORTHI, R. 2014. Rendering glints on high-resolution normal-mapped specular surfaces. *ACM Transactions on Graphics (Proc. SIGGRAPH) 33*, 4, 116:1–116:9.