# Supplemental materials: Multiple-bounce Smith Microfacet BRDFs using the Invariance Principle

Yuang Cui*
Anhui Science and Technology University
China
yuangcui@outlook.com

Gaole Pan*
Nanjing University of Science and Technology
China
pangaole@njust.edu.cn

Jian Yang
Nanjing University of Science and Technology
China
csjyang@njust.edu.cn

Lei Zhang
The Hong Kong Polytechnic University
China
cslzhang@comp.polyu.edu.hk

Ling-Qi Yan
University of California Santa Barbara
USA
lingqi@cs.ucsb.edu

Beibei Wang†
Nankai University, Nanjing University of Science and Technology
China
beibei.wang@njust.edu.cn

## 1 RELATIONSHIP TO THE MODEL BY BITTERLI AND D'EON [Bitterli and d'Eon 2022]

Our model and the model by Bitterli and d'Eon [2022] are related. They have different formulations, but their formulations can be proved equivalent for specific bounces. In this section, we provide the formulation of these two models in Sec. 1.1 and then prove their equivalence for specific bounces (2 and 3) in Sec. 1.2.

### 1.1 Formulation of two models

*Our model.* We follow the multiple-bounce BRDF model by Wang et al. [2022], defining the contribution of a light path with vertex terms and segment terms. In our model, the contribution of a light path includes vertex terms and a segment term:

$$f(\bar{x}) = \left(\prod_{i=0}^{i=k-1} v_i\right) S_k(\mathrm{d}_0, \mathrm{d}_1, \ldots, \mathrm{d}_k), \tag{1}$$

where the segment term $S_k$ has the formulation, as also shown in the main text:

$$S_1(\mathrm{d}_0, \mathrm{d}_1) = \frac{1}{1 + \Lambda(-\mathrm{d}_0) + \Lambda(\mathrm{d}_1)},$$
$$S_k(\mathrm{d}_0, \ldots, \mathrm{d}_k) = \begin{cases} 0, \text{ if } \mathrm{d}_0.z > 0 \text{ or } \mathrm{d}_k.z < 0, \\ S_1(\mathrm{d}_0, \mathrm{d}_k)\left(S_{k-1}(\mathrm{d}_0, \ldots, \mathrm{d}_{k-1}) + \right. \\ \left. S_{k-1}(\mathrm{d}_1, \ldots, \mathrm{d}_k)\right), \text{ otherwise.} \end{cases} \tag{2}$$

*The model by Bitterli and d'Eon [2022].* Bitterli and d'Eon [2022] perform a preintegration of the height component, using an explicit formulation of the height distribution as a hyperexponential distribution, resulting in an angular domain-only formulation. Their final formulation includes a phase function and a $p_{\text{exit}}$:

$$f(\bar{x}) = \left(\prod_{i=0}^{i=k-1} f_p(\mathrm{d}_i, \mathrm{d}_{i+1})\right) p_{\text{exit}}(\mathrm{d}_0, \mathrm{d}_1, \ldots, \mathrm{d}_k), \tag{3}$$

where $f_p$ is the phase function of the scattering, as defined by Heitz et al. [2016]:

$$f_p = \frac{F(\omega_i, \omega_h) D_{\omega_i}(\omega_h)}{4|\omega_h \cdot \omega_i|}. \tag{4}$$

The $p_{\text{exit}}$ is the probability of a photon exiting the medium, conditioned on the directions it takes after each collision, defined as:

$$p_{\text{exit}}(\mathrm{d}_0, \mathrm{d}_1, \ldots, \mathrm{d}_k) = \sum_{j=1}^{N_i} \frac{a_{i,j}}{b_{i,j} + \Lambda(\mathrm{d}_k)}. \tag{5}$$

$N_i, a_{i,j}, b_{i,j}$ are the number and coefficients of exponentials in the i-th height distribution, where:

$$\begin{aligned} & a_{1,1} = b_{1,1} = \Lambda(\mathrm{d}_0), N_1 = 1, \\ & a^{\downarrow}_{i+1,j} = \begin{cases} a_{i,j}\frac{\Lambda(\mathrm{d}_i)}{\Lambda(\mathrm{d}_i) - b_{i,j}}, & \text{if } j < N_{i+1}, \\ \sum_{j=1}^{N_i} -a_{i+1,j}, & \text{else,} \end{cases} \\ & b^{\downarrow}_{i+1,j} = \begin{cases} b_{i,j}, & \text{if } j < N_{i+1}, \\ \Lambda(\mathrm{d}_i), & \text{else,} \end{cases} \quad \text{and } N^{\downarrow}_{i+1} = N_i + 1, \\ & a^{\uparrow}_{i+1,j} = a_{i,j}\frac{\Lambda(\mathrm{d}_i)}{b_{i,j} + \Lambda(\mathrm{d}_i)}, b^{\uparrow}_{i+1,j} = b_{i,j} \text{ and } N^{\uparrow}_{i+1} = N_i. \end{aligned} \quad ?? \tag{6}$$

The superscripted arrow is the direction of the i-th bounce.

Note that, their model model might have singularities, which come from the minus of two $\Lambda$ functions.

*Comparison between two models.* This $p_{\text{exit}}$ from Bitterli and d'Eon [2022] is similar but not equivalent to our segment term, since some terms in their $p_{\text{exit}}$ are included in our vertex term. The relationship between our path segment term and the $p_{\text{exit}}$ is:

$$p_{\text{exit}}(\mathrm{d}_0, \mathrm{d}_1, \ldots, \mathrm{d}_k) = \left(\prod_{i=0}^{i=k-1} |\Lambda(\mathrm{d}_i)|\right) S_k(\mathrm{d}_0, \mathrm{d}_1, \ldots, \mathrm{d}_k). \tag{7}$$

### 1.2 Equivalence derivation between two models

We find that Bitterli and d'Eon [2022]'s model and ours are equivalent for specific bounces. Here, we provide explicit derivations of equivalence between their model and ours for specific paths with

*Contribute equally. Research done when Yuang Cui was an intern at Nanjing University of Science and Technology.
†Corresponding author.

**Algorithm 1:** SegmentTerm

```
class SegmentTerm($\Lambda_o$) // Initialize $\Lambda_o$
    def $m \leftarrow 1$
    def $N \leftarrow 0$
    def $e[], g[], l[]$
    function addBounce($\Lambda_k$)
        if $\Lambda_k < 0$ then // Downwards
            $N \leftarrow N + 1$
            $l[N] \leftarrow |\Lambda_k|$
            $e[N] \leftarrow \frac{1}{\Lambda_o + |\Lambda_k|}$
            $g[N] \leftarrow 0$
            $m \leftarrow m \cdot e[N]$// cache the results when
              sampled rays are only downwards
        else // Upwards
            if $m = 0$ then
                $g[N] \leftarrow \frac{g[N]}{\Lambda_k + l[N]}$
            else // Initialize g
                $g[N] \leftarrow \frac{1}{\Lambda_k + l[N]}$
                $m \leftarrow 0$
            for $i \leftarrow N - 1...1$ do
                $g[i] \leftarrow \frac{g[i] + g[i+1]}{\Lambda_k + l[i]}$
    function getS$_k$()
        if $m \neq 0$ then // Return results if cached
            return $m$
        $s \leftarrow 0$
        for $i \leftarrow N...1$ do
            $s \leftarrow e[i] \cdot (s + g[i])$
        return $s$
```

2 or 3 bounces. However, it is non-trivial to generalize to arbitrary bounces.

*Equivalence derivation for light path with two bounces.*

$$p_{\text{exit}}(\mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_2) = \sum_{j=1}^{N_i} \frac{a_{i,j}}{b_{i,j} + \Lambda(\mathrm{d}_k)}$$

$$= \frac{|\Lambda(\mathrm{d}_0)| \cdot \frac{|\Lambda(\mathrm{d}_1)|}{|\Lambda(\mathrm{d}_1)| - |\Lambda(\mathrm{d}_0)|}}{\Lambda(\mathrm{d}_2) + |\Lambda(\mathrm{d}_0)|} + \frac{-|\Lambda(\mathrm{d}_0)| \cdot \frac{|\Lambda(\mathrm{d}_1)|}{|\Lambda(\mathrm{d}_1)| - |\Lambda(\mathrm{d}_0)|}}{\Lambda(\mathrm{d}_2) + |\Lambda(\mathrm{d}_1)|}$$

$$= \frac{|\Lambda(\mathrm{d}_0)| \cdot |\Lambda(\mathrm{d}_1)|}{|\Lambda(\mathrm{d}_1)| - |\Lambda(\mathrm{d}_0)|} \left( \frac{1}{\Lambda(\mathrm{d}_2) + |\Lambda(\mathrm{d}_0)|} - \frac{1}{\Lambda(\mathrm{d}_2) + |\Lambda(\mathrm{d}_1)|} \right)$$

$$= \frac{|\Lambda(\mathrm{d}_0)| \cdot |\Lambda(\mathrm{d}_1)|}{|\Lambda(\mathrm{d}_1)| - |\Lambda(\mathrm{d}_0)|} \left( \frac{\Lambda(\mathrm{d}_2) + |\Lambda(\mathrm{d}_1)| - (\Lambda(\mathrm{d}_2) + |\Lambda(\mathrm{d}_0)|)}{(\Lambda(\mathrm{d}_2) + |\Lambda(\mathrm{d}_0)|)(\Lambda(\mathrm{d}_2) + |\Lambda(\mathrm{d}_1)|)} \right)$$

$$= \frac{|\Lambda(\mathrm{d}_0)| \cdot |\Lambda(\mathrm{d}_1)|}{|\Lambda(\mathrm{d}_1)| - |\Lambda(\mathrm{d}_0)|} \left( \frac{|\Lambda(\mathrm{d}_1)| - |\Lambda(\mathrm{d}_0)|}{(\Lambda(\mathrm{d}_2) + |\Lambda(\mathrm{d}_0)|)(\Lambda(\mathrm{d}_2) + |\Lambda(\mathrm{d}_1)|)} \right)$$

$$= \frac{|\Lambda(\mathrm{d}_0)| \cdot |\Lambda(\mathrm{d}_1)|}{(\Lambda(\mathrm{d}_2) + |\Lambda(\mathrm{d}_0)|)(\Lambda(\mathrm{d}_2) + |\Lambda(\mathrm{d}_1)|)}$$

$$= \left( \prod_{i=0}^{i=k-1} |\Lambda(\mathrm{d}_i)| \right) S_k(\mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_2).$$

**Algorithm 2:** Sample

```
Result: weight $w$ and direction $\mathrm{d}_k$
$p \leftarrow 1$ // pdf
$k \leftarrow 0$ // bounce
$w \leftarrow 1$ // weight
$\mathrm{d}_0 \leftarrow -\omega_i$
while true do
    ($\mathrm{d}_k$,$p_k$,$w_k$) $\leftarrow$ sample($\mathrm{d}_{k-1}$)
    $w \leftarrow w \cdot w_k$
    $p \leftarrow p \cdot p_k$
    if $\mathrm{d}_k.z > 0$ then
        $G_1 \leftarrow \frac{1}{|\Lambda(\mathrm{d}_k)|}$
        if rand() $< G_1$ then // the ray continues
          the tracing with $G_1$ as the probablity
            $p \leftarrow p \cdot G_1$
            break;
        else
            $p \leftarrow p \cdot (1 - G_1)$
    $k \leftarrow k + 1$
    if $k \geq maxBounce$ then
        break;
$w \leftarrow w \cdot S_k(\mathrm{d}_0, ..., \mathrm{d}_k)$
$w \leftarrow \frac{w}{p}$
```

**Algorithm 3:** Eval

```
function eval($\omega_i, \omega_o$)
    SegmentTerm $s(\Lambda(\omega_o))$// initializing
    $f \leftarrow 0$// result
    $k \leftarrow 0$ // bounce
    $w \leftarrow 1$// weight
    $\mathrm{d}_0 \leftarrow -\omega_i$
    while true do
        $s$.addBounce($\Lambda(\mathrm{d}_k)$)
        $k \leftarrow k + 1$
        if $k >= rrDepth$ then
            if rand() $> q$ then
                break;
            $w \leftarrow \frac{w}{q}$// russian roulette
        $f \leftarrow f + w \cdot v(\mathrm{d}_k, \omega_o) \cdot s$.getS$_k$()
        if $k \geq maxBounce$ then
            break;
        $\mathrm{d}_{k+1}$,$p_k$ $\leftarrow$ sample($\mathrm{d}_k$)
        $w \leftarrow \frac{w \cdot v(\mathrm{d}_k, \mathrm{d}_{k+1})}{p_k}$
    return $f$
```

*Equivalence derivation for light path with three bounces.*

$$p_{\text{exit}}(\mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_2, \mathrm{d}_3) = \sum_{j=1}^{N_i} \frac{a_{i,j}}{b_{i,j} + \Lambda(\mathrm{d}_k)}$$

$$
\begin{aligned}
&\frac{|\Lambda(\mathrm{d}_0)| \cdot \frac{|\Lambda(\mathrm{d}_1)|}{|\Lambda(\mathrm{d}_1)|-|\Lambda(\mathrm{d}_0)|} \cdot \frac{\Lambda(\mathrm{d}_2)}{\Lambda(\mathrm{d}_2)+|\Lambda(\mathrm{d}_0)|}}{\Lambda(\mathrm{d}_3) + |\Lambda(\mathrm{d}_0)|} \\
&+\frac{-|\Lambda(\mathrm{d}_0)| \cdot \frac{|\Lambda(\mathrm{d}_1)|}{|\Lambda(\mathrm{d}_1)|-|\Lambda(\mathrm{d}_0)|} \cdot \frac{\Lambda(\mathrm{d}_2)}{\Lambda(\mathrm{d}_2)+|\Lambda(\mathrm{d}_1)|}}{\Lambda(\mathrm{d}_3) + |\Lambda(\mathrm{d}_1)|} \\
&=\frac{|\Lambda(\mathrm{d}_0)| \cdot |\Lambda(\mathrm{d}_1)| \cdot |\Lambda(\mathrm{d}_2)|}{(|\Lambda(\mathrm{d}_1)| - |\Lambda(\mathrm{d}_0)|)\left(|\Lambda(\mathrm{d}_1)| + \Lambda(\mathrm{d}_2)\right)} \\
&\cdot\frac{1}{\left(|\Lambda(\mathrm{d}_1)| + \Lambda(\mathrm{d}_3)\right)\left(|\Lambda(\mathrm{d}_0)| + \Lambda(\mathrm{d}_2)\right)\left(|\Lambda(\mathrm{d}_0)| + \Lambda(\mathrm{d}_3)\right)} \\
&\cdot\Big(|\Lambda(\mathrm{d}_0)| \cdot \Lambda(\mathrm{d}_3) - |\Lambda(\mathrm{d}_0)| \cdot \Lambda(\mathrm{d}_2) + |\Lambda(\mathrm{d}_0)|^2 \\
&- |\Lambda(\mathrm{d}_1)| \cdot \Lambda(\mathrm{d}_3) + |\Lambda(\mathrm{d}_1)| \cdot \Lambda(\mathrm{d}_2) - |\Lambda(\mathrm{d}_1)|^2\Big) \\
&= \frac{|\Lambda(\mathrm{d}_0)| \cdot |\Lambda(\mathrm{d}_1)| \cdot \Lambda(\mathrm{d}_2)}{\left(|\Lambda(\mathrm{d}_1)| + \Lambda(\mathrm{d}_2)\right)\left(|\Lambda(\mathrm{d}_0)| + \Lambda(\mathrm{d}_3)\right)} \\
&\cdot\left(\frac{1}{|\Lambda(\mathrm{d}_0)| + \Lambda(\mathrm{d}_2)} + \frac{1}{|\Lambda(\mathrm{d}_1)| + \Lambda(\mathrm{d}_3)}\right) \\
&=\left(\prod_{i=0}^{i=k-1} |\Lambda(\mathrm{d}_i)|\right) S_k(\mathrm{d}_0, \mathrm{d}_1, \mathrm{d}_2, \mathrm{d}_3).
\end{aligned}
$$

## 2 IMPLEMENTATION DETAILS

In this section, we provide the pseudo-code for our models (both BRDF evaluation and sample) and show the explicit implementation difference between Bitterli and d'Eon [2022] and our model.

### 2.1 Implementation details of our model

*Evaluation.* We provide the pseudo-code of our BRDF evaluation with the unidirectional estimator in Alg. 3 and further show the detailed implementation of our segment term using dynamic programming in Alg. 1.

*Sample.* We show the multiple-bounce BRDF sample in Alg. 2. At each bounce, we sample the VNDF to get the outgoing direction and then compute the masking function ($G_1$ function) of the sampled ray to decide whether to exit the microgeometry. Otherwise, we treat the sampled direction as the incoming direction and continue sampling until the ray leaves the surface. After getting such a light path, we compute the path contribution by evaluating the light path divided by the PDF. Using the masking function of the last sampled ray as the exit probability enables satisfying results with less time cost, while using the entire sampled path to compute the exit probability introduces a large time overhead.

*BDPT.* The bidirectional estimator is the same as Wang et al. [2022], except for the segment term. We did not provide the results of Bitterli and d'Eon [2022] (BDPT), since we can not get the correct results after implementing their pseudo-code. We also find that their algorithm does not provide an expected result (single-bounce microfacet BRDF, even set the path length as 2).

### 2.2 Implementation comparison between our model and Bitterli and d'Eon

In this section, we compare the implementation details between our segment term and the $p_{\mathrm{exit}}$ by Bitterli and d'Eon [2022]. We have highlighted the main parts of our algorithm in Alg. 4 and their algorithm in Alg. 5.

**Algorithm 4:** SegmentTerm (Ours)

```
class SegmentTerm($\Lambda_o$)
    $N \leftarrow 0$
    $m \leftarrow 1$
    def $e[], g[], l[]$
    function addBounce($\Lambda_k$)
        if $\Lambda_k < 0$ then
            $N \leftarrow N + 1$
            $l[N] \leftarrow |\Lambda_k|$
            $e[N] \leftarrow \frac{1}{\Lambda_o + |\Lambda_k|}$
            $g[N] \leftarrow 0$
            $m \leftarrow m \cdot e[N]$
        else
            if $m = 0$ then
                $g[N] \leftarrow \frac{g[N]}{\Lambda_k + l[N]}$
            else
                $g[N] \leftarrow \frac{1}{\Lambda_k + l[N]}$
                $m \leftarrow 0$
            for $i \leftarrow N - 1 \ldots 1$ do
                $g[i] \leftarrow \frac{g[i] + g[i+1]}{\Lambda_k + l[i]}$
    function getS$_k$()
        if $m \neq 0$ then
            return $m$
        $s \leftarrow 0$
        for $i \leftarrow N \ldots 1$ do
            $s \leftarrow e[i] \cdot (s + g[i])$
        return $s$
```

The different formulations of the two algorithms lead to different performances. In the end, our method has less time cost when rendered with an equal sample rate.

## 3 MORE RESULTS

As shown in Figure 1, our PDF is less accurate for anisotropic BRDFs than for isotropic media, since our mapping function averages the roughness for anisotropic BRDFs. Even though, our PDF is still better than the single bounce + the Lambertian term approximation.

## REFERENCES

Benedikt Bitterli and Eugene d'Eon. 2022. A Position-Free Path Integral for Homogeneous Slabs and Multiple Scattering on Smith Microfacets. *Computer Graphics Forum* 41, 4 (2022), 93–104. https://doi.org/10.1111/cgf.14589

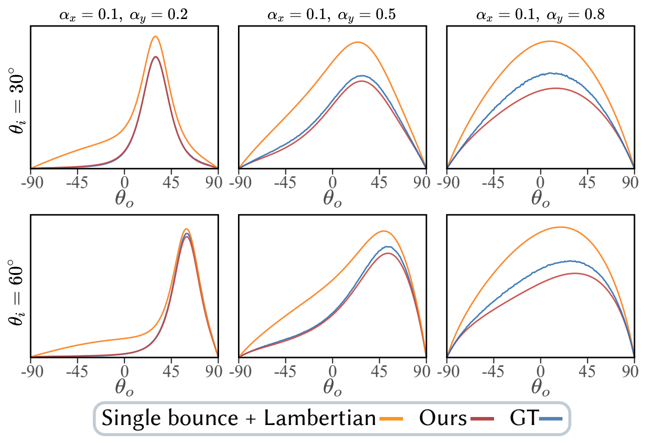

**Figure 1: Comparison between our improved PDF, the previous solution (single bounce + the Lambertian term) by Heitz et al. [2016] and the ground truth using the multiple-bounce BRDF reflectance.**

**Algorithm 5:** HeightDistribution (Bitterli and d'Eon)

```
class HeightDistribution($\Lambda_o$)
    def $N, a[], b[]$
    function addBounce($\Lambda_k$)
        if $N = 0$ then
            $a[1] \leftarrow \Lambda_k$
            $b[1] \leftarrow \Lambda_k$
            $N \leftarrow N + 1$
        else if $\Lambda_k < 0$ then
            for $i \leftarrow 1...N$ do
                $a[i] \leftarrow a[i] \frac{\Lambda_k}{\Lambda_k - b[i]}$
            $a[N+1] \leftarrow 0$
            for $i \leftarrow 1...N$ do
                $a[N+1] \leftarrow a[N+1] - a[i]$
            $b[N+1] \leftarrow \Lambda_k$
            $N \leftarrow N + 1$
        else
            for $i \leftarrow 1...N$ do
                $a[i] \leftarrow a[i] \frac{\Lambda_k}{\Lambda_k + b[i]}$
            $a[N+1] \leftarrow 0$
            $b[N+1] \leftarrow -\Lambda_k$
    function getP$_{\text{exit}}$()
        $s \leftarrow 0$
        for $i \leftarrow 1...N$ do
            $s \leftarrow s + \frac{a[i]}{\Lambda_o + b[i]}$
        return $s$
```

Eric Heitz, Johannes Hanika, Eugene d'Eon, and Carsten Dachsbacher. 2016. Multiple-Scattering Microfacet BSDFs with the Smith Model. *ACM Trans. Graph.* 35, 4, Article 58 (July 2016), 14 pages.

Beibei Wang, Wenhua Jin, Jiahui Fan, Jian Yang, Nicolas Holzschuch, and Ling-Qi Yan. 2022. Position-Free Multiple-Bounce Computations for Smith Microfacet BSDFs. *ACM Trans. Graph.* 41, 4, Article 134 (jul 2022), 14 pages.