1 Derivation of the Smith Normalization Factor

This section derives closed-form solutions for the normalization factor:

\[
\frac{G(i \cdot m)}{H(i)} = \frac{|i \cdot n|}{\int_{S^2} D(\omega) \max(i \cdot \omega, 0) d\omega} \chi^+(i \cdot m). \tag{1}
\]

The NDF \(D(m)\) is expressed using a slope distribution \(P_22(x(m), y(m))\) as follows:

\[
D(m(m \cdot n)) = P_22(x(m), y(m)) \chi^+(m \cdot n) \left\| \frac{\partial [x(m), y(m)]}{\partial m} \right\|,
\]

where \(\|\partial [x(m), y(m)]/\partial m\| = 1/(m \cdot n)^3\) is the Jacobian for the transformation between the microfacet normal \(m\) and its slope \([x(m), y(m)]\). Using this slope distribution, \(\int_{S^2} D(\omega) \max(i \cdot \omega, 0) d\omega\) can be rewritten into the following slope-space integral [Hei14]:

\[
\int_{S^2} D(\omega) \max(i \cdot \omega, 0) d\omega = \int_{-\infty}^{\infty} (\cos \theta - x \sin \theta) P_2(x) dx,
\]

where \(\cos \theta = i \cdot n = i_z, \) and \(P_2(x) = \int_S P_22(x, y) dy\).

Smith–Beckmann Model. For the GGX NDF [WMLT07], the slope-space distribution is the following bivariate elliptical distribution:

\[
P_22(x, y) = \frac{1}{\pi \sqrt{|A|}} \left(\frac{x^2 + y^2}{|A^{-1}[(x, y)'] + 1|}\right)^{\alpha},
\]

where \(A\) is a positive semi-definite \(2\times2\) matrix whose eigenvalues are \([\alpha_x^2, \alpha_y^2]\), and eigenvectors are tangent and binormal vectors. For this distribution, \(P_2(x)\) is given by

\[
P_2(x) = \int_{-\infty}^{\infty} P_22(x, y) dy = \frac{2\alpha_x}{2(\alpha_x^2 + x^2)^{\frac{3}{2}}},
\]

where \(\alpha_x^2 = (\alpha_x^2 i_z^2 + \alpha_y^2 j_z^2)/(i_z^2 + j_z^2)\) is the squared roughness projected onto the x-axis. Substituting this \(P_2(x)\) into Eq. (2), we yield

\[
\int_{S^2} D(\omega) \max(i \cdot \omega, 0) d\omega = \int_{-\infty}^{\infty} \frac{2\alpha_x (\cos \theta - x \sin \theta)}{2(\alpha_x^2 + x^2)^{\frac{3}{2}}} dx = \frac{\cos \theta + \sqrt{\alpha_x^2 \sin^2 \theta + \cos^2 \theta}}{2}.
\]

Since \(\cos \theta = i_z\) and \(\alpha_x^2 \sin^2 \theta = \alpha_x^2 i_z^2 + \alpha_y^2 j_z^2\), we obtain

\[
\int_{S^2} D(\omega) \max(i \cdot \omega, 0) d\omega = \frac{i_z + \sqrt{\alpha_x^2 i_z^2 + \alpha_y^2 j_z^2 + k_z^2}}{2}.
\]

Substituting this equation into Eq. (1), we yield the normalization factor for the Smith–GGX model:

\[
\frac{G(i \cdot m)}{H(i)} = \frac{2|i_z|}{i_z + \sqrt{\alpha_x^2 i_z^2 + \alpha_y^2 j_z^2 + k_z^2}} \chi^+(i \cdot m).
\]

Smith–Beckmann Model. The slope-space distribution of the Beckmann NDF [BS63] is a bivariate Gaussian:

\[
P_{22}(x, y) = \frac{1}{\pi \sqrt{|A|}} \exp \left(-\frac{\|x, y\|}{A^{-1}[(x, y)']}ight).
\]

For this distribution, \(P_2(x)\) is a 1D Gaussian distribution:

\[
P_2(x) = \int_{-\infty}^{\infty} P_{22}(x, y) dy = \frac{1}{\sqrt{2\pi}} \exp \left(-\frac{x^2}{\alpha^2}\right).
\]

Substituting this \(P_2(x)\) into Eq. (2), we yield

\[
\int_{S^2} D(\omega) \max(i \cdot \omega, 0) d\omega = \int_{-\infty}^{\infty} \frac{\cos \theta - x \sin \theta}{\sqrt{2\pi}} \exp \left(-\frac{x^2}{\alpha^2}\right) dx = \frac{\cos \theta + \frac{\sqrt{\alpha^2 \sin^2 \theta + \cos^2 \theta}}{\alpha}}{2}.
\]

\[
= \frac{\alpha}{2} \text{erf} \left(-\frac{i_z}{\sqrt{2}}\right) + \sqrt{\frac{\alpha}{2}} \exp \left(-\frac{i_z^2}{4}\right),
\]

where \(B = \alpha^2 \sin^2 \theta = \alpha_x^2 i_z^2 + \alpha_y^2 j_z^2\). Substituting this equation into Eq. (1), we yield the normalization factor for the Smith–Beckmann model:

\[
\frac{G(i \cdot m)}{H(i)} = \frac{2|i_z|}{i_z \text{erf} \left(-\frac{i_z}{\sqrt{2}}\right) + \sqrt{\frac{\alpha}{2}} \exp \left(-\frac{i_z^2}{4}\right)} \chi^+(i \cdot m).
\]

2 The V-Cavity Model for Shading Normals

V-cavity microsurface is formed by a set of symmetric V-grooves (Fig. 1). Similar to the Smith model, we assume that V-cavity microfacets are single sided. This assumption does not affect the masking function for frontfac ing shading normals, because back-facing microfacets are fully masked by front-facing microfacets (Fig. 1b). On the other hand, our assumption makes front-facing microfacets visible from below the horizon (Figs. 1c and 1d).

2.1 Masking Function

Previous work considered two masking configurations: one in which both sides of V-cavity are front-facing and fully visible (Fig. 1a), and one in which only one side of V-cavity is front-facing (Fig. 1b). These two configurations are expressed in a single formula:

\[
G(i \cdot m) = \min \left(\frac{2|m \cdot n| |i \cdot n|}{|i \cdot m|}, 1\right) \chi^+(i \cdot m).
\]
front-facing microfacets (Fig. 1c). The second one is that only one side of V-cavity is front-facing and not masked (Fig. 1d). The third one is that both sides of V-cavity are back-facing and fully invisible (Fig. 1e). Unlike the Smith model, the microsurface visibility for back-facing shading normals is given from these three configurations, and it results in the same form as Eq. (3).

2.2 Hit Probability

The hit probability \( H(i) \) is obtained from the masking function (Eq. 3) and NDF. Closed-form solutions for this hit probability is available for the GGX and Beckmann NDFs as follows:

For the GGX NDF

\[
H(i) = \begin{cases} 
1 & \text{if } \mathbf{i} \cdot \mathbf{n} \geq 0 \\
\frac{1}{1 + \frac{\sqrt{\alpha^2 + \theta^2 + 2\mathbf{i} \cdot \mathbf{n}}}{\sqrt{\alpha^2 + \theta^2 + 2\mathbf{i} \cdot \mathbf{n}}} + \frac{\sqrt{\alpha^2 + \theta^2 + 2\mathbf{i} \cdot \mathbf{n}}}{\sqrt{\alpha^2 + \theta^2 + 2\mathbf{i} \cdot \mathbf{n}}} + \frac{\mathbf{i} \cdot \mathbf{n} - \mathbf{i} \cdot \mathbf{n}}{2} \} & \text{otherwise} 
\end{cases}
\]

For the Beckmann NDF

\[
H(i) = \begin{cases} 
1 & \text{if } \mathbf{i} \cdot \mathbf{n} \geq 0 \\
\left( 1 + \frac{\exp(-\alpha^2) - \exp(-\theta^2)}{\exp(-\theta^2)} \right) & \text{otherwise} 
\end{cases}
\]

where \( u = \frac{\mathbf{n}}{\sqrt{\alpha^2 + \theta^2}} \).

2.3 VNDF Sampling Routine

While V-cavity VNDF sampling for front-facing shading normals is independent from NDF models [Hd14], sampling for back-facing shading normals depends on an NDF model. The PDF for back-facing shading normals has different forms depending on the slope \( 3 \cos \theta \) as follows:

\[
p(m; i) = \begin{cases} 
\frac{\int D(m)(m \mathbf{n})}{H(i)} & \text{if } x(m) < 3 \cos \theta \\
\frac{\int D(m)(m \mathbf{n})}{H(i)} & \text{if } 3 \cos \theta \leq x(m) < 3 \cos \theta \\
0 & \text{otherwise} 
\end{cases}
\]

 Therefore, we first choose stochastically whether a sample microfacet slope \( x(m) \) exceeds \( 3 \cos \theta \) or not. Then, we sample a microfacet normal according to the chosen PDF form. The probability that \( x(m) < 3 \cos \theta \) is given by the integral of the PDF in this case as follows:

\[
\int_{-\infty}^{\infty} x'(3 \cos \theta - \theta) p(m; i) \, dx = \frac{2}{H(i)} \int_{-\infty}^{\infty} P_2(x) \, dx.
\]

If \( x(m) < 3 \cos \theta \), we sample a microfacet \( m \) according to \( D(m)(m \mathbf{n}) \) with the limited range \( x(m) \in (-\infty, 3 \cos \theta) \). For \( x(m) \geq 3 \cos \theta \), the PDF is proportional to the VNDF of the Smith model. Therefore, we sample a microfacet by limiting the range of Smith VNDF sampling [Hei18, Jak14] to \( x(m) \in [3 \cos \theta, \cot \theta] \) for this case.

Listing 1 shows our VNDF sampling routine for the V-cavity model. For \( \text{FrontFacingProportion} \) (boundary, roughness) computes \( \int_{-\infty}^{3 \cos \theta} P_2(x) \, dx \) which is given by

\[
\int_{-\infty}^{3 \cos \theta} P_2(x) \, dx = 1 + \frac{k_i}{\sqrt{a_i^2 + k_i^2}}
\]

for the GGX NDF,

\[
\int_{-\infty}^{3 \cos \theta} P_2(x) \, dx = 1 + \frac{k_i}{\sqrt{a_i^2 + k_i^2}}
\]

for the Beckmann NDF.

where \( k_i, k_i, k_i \) is the normal of the boundary surface (i.e., boundary). For the GGX NDF, \( \text{SampleNormal} \) and \( \text{SampleUnmaskedNormal} \) routines are shown in Listings 2 and 3 which are based on a Smith-GGX VNDF sampling routine [Hei18]. For the Beckmann NDF, we employ slope-space sampling [Hd14] shown in Listings 4 and 5. To sample a slope by solving the inverse cumulative distribution function, we use the Newton’s method based on Jakob’s approach [Jak14]. Although the original Jakob’s sampling routine used the bisection method in addition to the Newton’s method, we omitted the bisection method similar to Mitsuba 2 [NDVZJ19]. Listing 6 shows our slope sampling routine using the Newton method.

Listing 1: HLSL-like pseudo code of our VNDF sampling for the V-cavity model.

```c
float3 SampleVisibleNormal(float u1, float u2, float u3, float2 dir) {
  if (dir.z <= 0.0) {
    // Existing V-cavity VNDF sampling for front-facing shading normals [Heitz and d’Eon 2014].
    float3 normal1 = SampleNormal(u1, u2, roughness, float2(0.0, 0.0), 1.0);
    float3 normal2 = {-normal1.x, -normal1.y, normal1.z};
    float a2 = max(dot(dir, normal2), 0.0);
    float3 boundary = {dir.x, dir.y, 3.0 * dir.z};
    float probability = a1 / (a1 + a2);
    return u3 < probability ? normal1 : normal2;
  } else {
    // Normal of the boundary surface between two PDF forms
    boundary = {dir.x, dir.y, 3.0 * dir.z};
    float3 boundary = {dir.x, dir.y, 3.0 * dir.z};
    float a2 = max(dot(dir, normal2), 0.0);
    float3 normal = normal1 / (a1 + a2);
    return u3 < probability ? normal : normal2;
  }
  // Probability that a sample slope is less than the boundary.
  float s = frontFacingProportion(boundary, roughness);
  float probability = 2.0 * s / HitProbability(dir);
  return SampleNormal((u1 + probability, u2, roughness, dir.xy, s));
}
```
Listing 2: Sampling according to \( p(m,i) \propto D(m)(m \cdot n) \) for the GGX NDF. To limit the sampling range, this implementation is based on the Smith–GGX VNDF sampling routine [Hei18], and changes the projection direction from an incident direction to the shading normal (written in red).

```c
float3 SampleNormal(float u1, float u2, float2 roughness, float2 dir, float s) {  // Stretch the incident direction.
    float2 d = dir * roughness;
    float3 sample = SampleNormal(u1, u2, roughness) * d;
    // Normalize sample.
    return normalize(sample);
}
```

Listing 3: Sampling according to \( p(m,i) \propto D(m)(i \cdot m) \) based on the Smith–GGX VNDF sampling routine [Hei18]. This implementation modifies the upper limit of the sampling range (written in red).

```c
float3 SampleUnmaskedNormal(float u1, float u2, float2 roughness, float3 dir) {  // Stretch and normalize the incident direction.
    float3 n = dir * roughness;
    // Compute the upper limit of the sampling range: \( \alpha = \frac{3 \cot(\theta)}{4} \).
    float limit = 3.0 * dir.z / sinTheta;
    // Stretch and normalize the incident direction.
    float3 d = normalize(float3(dir.xy * roughness, dir.z));
    // Transform the sampled slope.
    float2 n = \(-\alpha x + \alpha z \) y;
    // Unstretch and normalize the microfacet normal.
    return normalize(float3(n * roughness, 1.0));
}
```

Listing 4: Sampling according to \( p(m,i) \propto D(m)(m \cdot n) \) for the Beckmann NDF. Unlike the Box-Muller’s method, this implementation uses the inverse error function for each slope axis to limit the sampling range using \( s \) (written in red).

```c
float3 SampleNormal(float u1, float u2, float2 roughness, float2 dir, float s) {  // Stretch the incident direction.
    float2 d = dir * roughness;
    // Sample a slope.
    float x = -erfinv(2.0 * u1 - 1.0);
    // Build an orthonormal basis.
    float lensq = d.x * d.x + d.y * d.y;
    float2 axisX = lensq <- 0.0 7 d / sqrt(lensq) : float2(1.0, 0.0);
    float2 axisY = \(-\alpha x, \alpha z \) y;
    // Transform the sampled slope.
    float2 n = \(-\alpha x \) x + \alpha z y;
    // Unstretch and normalize the microfacet normal.
    return normalize(float3(n * roughness, 1.0));
}
```

Listing 5: Sampling according to \( p(m,i) \propto D(m)(i \cdot m) \) based on the Smith–Beckmann VNDF sampling routine [Hd14]. This implementation limits the sampling range using \( \text{slopeMin} \) (written in red).

```c
float3 SampleUnmaskedNormal(float u1, float u2, float2 roughness, float4 dir) {  // Stretch and normalize the incident direction.
    float d = normalize(float3(dir.xy * roughness, dir.zz));
    // Compute a lower limit of the sampling range: \( 3 \cot(\theta) \).
    float sinTheta = length(d.z);
    float slopeMin = 3.0 * d.z / sinTheta;
    // Sample a slope.
    float y = -erfinv(2.0 * u2 - 1.0);
    // Build an orthonormal basis.
    float lensq = d.x * d.x + d.y * d.y;
    float2 axisX = lensq <- 0.0 7 d / sqrt(lensq) : float2(1.0, 0.0);
    float2 axisY = \(-\alpha x, \alpha z \) y;
    // Transform the sampled slope.
    float2 n = \(-\alpha x \) x + \alpha z y;
    // Unstretch and normalize the microfacet normal.
    return normalize(float3(n * roughness, 1.0));
}
```

Listing 6: HLSL-like pseudo code of our visible slope sampling for the Beckmann NDF. The main difference from the existing sampling routine is specification of the lower limit written in red.

```c
float3 SampleVisibleSlope(float u1, float u2, float sinTheta, float d.z, float slopeMin) {  // Compute the upper limit of \( \cot(\theta) \).
    float slopeMax = \cosTheta / \sinTheta;
    // Compute a lower limit of the sampling range: \( 3 \cot(\theta) \).
    float slopeMin = 3.0 * d.z / sinTheta;
    // Sample a slope.
    float y = -erfinv(2.0 * u2 - 1.0);
    // Build an orthonormal basis.
    float lensq = d.x * d.x + d.y * d.y;
    float2 axisX = lensq <- 0.0 7 d / sqrt(lensq) : float2(1.0, 0.0);
    float2 axisY = \(-\alpha x, \alpha z \) y;
    // Transform the sampled slope.
    float2 n = \(-\alpha x \) x + \alpha z y;
    // Unstretch and normalize the microfacet normal.
    return normalize(float3(n * roughness, 1.0));
}
```
3 Experimental Results

Fig. 2 shows rendering results and visualizations of the error and the VNDF integral for the Smith–Beckmann BRDF (2a), V-cavity–GGX BRDF (2b), and V-cavity–Beckmann BRDF (2c). To compute sample directions for the Smith–Beckmann model, we employ the Newton’s method based on Jakob’s sampling routine [Jak14]. Our normalization avoids a brightening bias caused by the VNDF integral being less than one.

Acknowledgments

Normal maps are generated from Crytek Sponza’s textures downloaded from M. McGuire’s Computer Graphics Archive. We would like to thank F. Meinl and E. Bischoff for the Crytek Sponza model.

References


