Unbiased VNDF Sampling for Backfacing Shading Normals (Supplementary Document)

Yusuke Tokuyoshi, AMD Japan Ltd.

1 Derivation of the Smith Normalization Factor

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

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

The NDF $D(\mathbf{m})$ is expressed using a slope distribution $P_{22}(x(\mathbf{m}), y(\mathbf{m}))$ as follows:

$$D(\mathbf{m})(\mathbf{m}\cdot\mathbf{n}) = P_{22}(x(\mathbf{m}), y(\mathbf{m}))\chi^+(\mathbf{m}\cdot\mathbf{n})\left\|\frac{\partial[x(\mathbf{m}), y(\mathbf{m})]}{\partial\mathbf{m}}\right\|,$$

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

$$\int_{\mathcal{S}^2} D(\boldsymbol{\omega})\max(\mathbf{i}\cdot\boldsymbol{\omega},0)\mathrm{d}\boldsymbol{\omega} = \int_{-\infty}^{\cot\theta}(\cos\theta - x\sin\theta)P_2(x)\mathrm{d}x, \tag{2}$$

where $\cos\theta = \mathbf{i}\cdot\mathbf{n} = i_z$, and $P_2(x) = \int_{\mathbb{R}} P_{22}(x,y)\mathrm{d}y$.

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

$$P_{22}(x,y) = \frac{1}{\pi\sqrt{\|\mathbf{A}\|}\left([x,y]\mathbf{A}^{-1}[x,y]^\top + 1\right)^2},$$

where $\mathbf{A}$ is a positive semi-definite 2×2 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

$$P_2(x) = \int_{\mathbb{R}} P_{22}(x,y)\mathrm{d}y = \frac{\alpha^2}{2(\alpha^2+x^2)^{\frac{3}{2}}},$$

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

$$\int_{\mathcal{S}^2} D(\boldsymbol{\omega})\max(\mathbf{i}\cdot\boldsymbol{\omega},0)\mathrm{d}\boldsymbol{\omega} = \int_{-\infty}^{\cot\theta}\frac{\alpha^2(\cos\theta - x\sin\theta)}{2(\alpha^2+x^2)^{\frac{3}{2}}}\mathrm{d}x = \frac{\cos\theta + \sqrt{\alpha^2\sin^2\theta + \cos^2\theta}}{2}.$$

Since $\cos\theta = i_z$ and $\alpha^2\sin^2\theta = \alpha_x^2 i_x^2 + \alpha_y^2 i_y^2$, we obtain

$$\int_{\mathcal{S}^2} D(\boldsymbol{\omega})\max(\mathbf{i}\cdot\boldsymbol{\omega},0)\mathrm{d}\boldsymbol{\omega} = \frac{i_z + \sqrt{\alpha_x^2 i_x^2 + \alpha_y^2 i_y^2 + i_z^2}}{2}.$$

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

$$\boxed{\frac{G(\mathbf{i},\mathbf{m})}{H(\mathbf{i})} = \frac{2|i_z|}{i_z + \sqrt{\alpha_x^2 i_x^2 + \alpha_y^2 i_y^2 + i_z^2}}\chi^+(\mathbf{i}\cdot\mathbf{m})}.$$

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

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

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

$$P_2(x) = \int_{\mathbb{R}} P_{22}(x,y)\mathrm{d}y = \frac{1}{\sqrt{\pi\alpha^2}}\exp\left(-\frac{x^2}{\alpha^2}\right).$$

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

$$\begin{aligned}\int_{\mathcal{S}^2} D(\boldsymbol{\omega})\max(\mathbf{i}\cdot\boldsymbol{\omega},0)\mathrm{d}\boldsymbol{\omega} &= \int_{-\infty}^{\cot\theta}\frac{\cos\theta - x\sin\theta}{\sqrt{\pi\alpha^2}}\exp\left(-\frac{x^2}{\alpha^2}\right)\mathrm{d}x\\ &= \frac{\cos\theta\,\mathrm{erfc}\left(-\frac{\cot\theta}{\alpha}\right) + \frac{\alpha\sin\theta}{\sqrt{\pi}}\exp\left(-\frac{\cot^2\theta}{\alpha^2}\right)}{2}\\ &= \frac{i_z\mathrm{erfc}\left(-\frac{i_z}{\sqrt{B}}\right) + \sqrt{\frac{B}{\pi}}\exp\left(-\frac{i_z^2}{B}\right)}{2},\end{aligned}$$

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

$$\boxed{\frac{G(\mathbf{i},\mathbf{m})}{H(\mathbf{i})} = \frac{2|i_z|}{i_z\mathrm{erfc}\left(-\frac{i_z}{\sqrt{B}}\right) + \sqrt{\frac{B}{\pi}}\exp\left(-\frac{i_z^2}{B}\right)}\chi^+(\mathbf{i}\cdot\mathbf{m})}.$$

2 The V-Cavity Model for Shading Normals

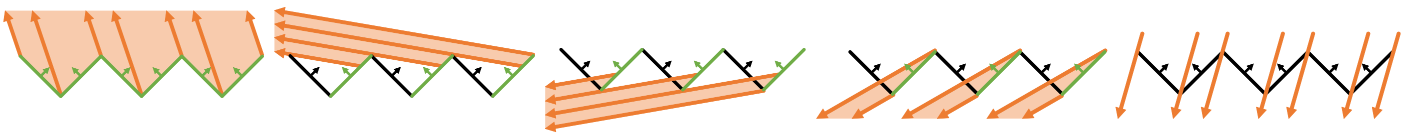
V-cavity microsurface is formed by a set of symmetric V-grooves (Fig. 1). Similar to the Smith model, we assume that V-cavity microfacets are single sided. This assumption does not affect the masking function for frontfacing shading normals, because backfacing microfacets are fully masked by frontfacing microfacets (Fig. 1b). On the other hand, our assumption makes frontfacing 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 frontfacing and fully visible (Fig. 1a), and one in which only one side of V-cavity is frontfacing (Fig. 1b). These two configurations are expressed in a single formula:

$$G(\mathbf{i},\mathbf{m}) = \min\left(\frac{2|\mathbf{m}\cdot\mathbf{n}||\mathbf{i}\cdot\mathbf{n}|}{|\mathbf{i}\cdot\mathbf{m}|}, 1\right)\chi^+(\mathbf{i}\cdot\mathbf{m}). \tag{3}$$

In addition to the above two configurations, we consider three more configurations for backfacing shading normals. The first one is that only one side of V-cavity is frontfacing and partially masked by other



(a) Fully visible (b) Partially masked (c) Partially masked (d) Unmasked and partially visible (e) Fully invisible

Figure 1: Masking configurations for our single-sided V-cavity model. Orange arrows are incident direction $\mathbf{i}$. Frontfacing microfacets (green) are visible not only from above the horizon (a, b), but also below the horizon (c, d).

frontfacing microfacets (Fig. 1c). The second one is that only one side of V-cavity is frontfacing and not masked (Fig. 1d). The third one is that both sides of V-cavity are backfacing and fully invisible (Fig. 1e). Unlike the Smith model, the microsurface visibility for backfacing 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 $H(\mathbf{i})$ 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

$$H(\mathbf{i}) = \begin{cases} 1 & \text{if } \mathbf{i}\cdot\mathbf{n} \geq 0 \\ 1 + \frac{\sqrt{\alpha_x^2 i_x^2 + \alpha_y^2 i_y^2 + 9 i_z^2} - \sqrt{\alpha_x^2 i_x^2 + \alpha_y^2 i_y^2 + i_z^2}}{2 i_z} & \text{otherwise} \end{cases}.$$

For the Beckmann NDF

$$H(\mathbf{i}) = \begin{cases} 1 & \text{if } \mathbf{i}\cdot\mathbf{n} \geq 0 \\ 1 + \frac{\exp(-9u^2) - \exp(-u^2)}{2\sqrt{\pi}u} + \frac{3\operatorname{erf}(3u) - \operatorname{erf}(u)}{2} & \text{otherwise} \end{cases},$$

where $u = \frac{i_z}{\sqrt{\alpha_x^2 i_x^2 + \alpha_y^2 i_y^2}}$.

2.3 VNDF Sampling Routine

While V-cavity VNDF sampling for frontfacing shading normals is independent from NDF models [Hd14], sampling for backfacing shading normals depends on an NDF model. The PDF for backfacing shading normals has different forms bordering on the slope $3\cot\theta$ as follows:

$$p(\mathbf{m};\mathbf{i}) = \begin{cases} \frac{2D(\mathbf{m})(\mathbf{m}\cdot\mathbf{n})}{H(\mathbf{i})} & \text{if } x(\mathbf{m}) < 3\cot\theta \\ \frac{D(\mathbf{m})(\mathbf{i}\cdot\mathbf{m})}{H(\mathbf{i})|\mathbf{i}\cdot\mathbf{n}|} & \text{if } 3\cot\theta \leq x(\mathbf{m}) < \cot\theta \\ 0 & \text{otherwise} \end{cases}. \tag{4}$$

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

$$\int_{S^2} \chi^+(3\cot\theta - x(\boldsymbol{\omega}))p(\boldsymbol{\omega};\mathbf{i})\mathrm{d}\boldsymbol{\omega} = \frac{2\int_{-\infty}^{3\cot\theta} P_2(x)\mathrm{d}x}{H(\mathbf{i})}. \tag{5}$$

If $x(\mathbf{m}) < 3\cot\theta$, we sample a microfacet $\mathbf{m}$ according to $D(\mathbf{m})(\mathbf{m}\cdot\mathbf{n})$ with the limited range $x(\mathbf{m}) \in (-\infty, 3\cot\theta)$. For $x(\mathbf{m}) \geq 3\cot\theta$, 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 $x(\mathbf{m}) \in [3\cot\theta, \cot\theta)$ for this case.

Listing 1 shows our VNDF sampling routine for the V-cavity model. *FrontfacingProportion(boundary, roughness)* computes $\int_{-\infty}^{3\cot\theta} P_2(x)\mathrm{d}x$ which is given by

$$\int_{-\infty}^{3\cot\theta} P_2(x)\mathrm{d}x = \frac{1}{2} + \frac{k_z}{2\sqrt{\alpha_x^2 k_x^2 + \alpha_y^2 k_y^2 + k_z^2}} \quad \text{for the GGX NDF,}$$

$$\int_{-\infty}^{3\cot\theta} P_2(x)\mathrm{d}x = \frac{1}{2}\operatorname{erfc}\left(-\frac{k_z}{\sqrt{\alpha_x^2 k_x^2 + \alpha_y^2 k_y^2}}\right) \quad \text{for the Beckmann NDF.}$$

where $[k_x, k_y, k_z] = \lfloor i_x, i_y, 3i_z \rceil$ is the normal of the boundary surface (i.e., *boundary*). For the GGX NDF, *SampleNormal* and *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.

```
float3 SampleVisibleNormal(float u1, float u2, float u3, float2
      roughness, float3 dir) {
 if (dir.z >= 0.0) {
  // Existing V-cavity VNDF sampling for frontfacing 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 a1 = max(dot(dir, normal1), 0.0);
  float a2 = max(dot(dir, normal2), 0.0);
  float probability = a1 / (a1 + a2);
  return u3 < probability ? normal1 : normal2;
 } else {
  // Normal of the boundary surface between two PDF forms
  float3 boundary = {dir.x, dir.y, 3.0 * dir.z};

  // Probability that a sample slope is less than the boundary.
  float s = FrontfacingProportion(boundary, roughness);
  float probability = 2.0 * s / HitProbability(dir);

  if (u1 < probability) {
   // Sampling according to a PDF proportional to D(m)*dot(m, n).
   return SampleNormal(u1 / probability, u2, roughness, dir.xy, s);
  } else {
   // Sampling according to a PDF proportional to D(m)*dot(i, m).
   return SampleUnmaskedNormal((u1 - probability) / (1.0 -
         probability), u2, roughness, dir);
  }
 }
}
```

Listing 2: Sampling according to $p(\mathbf{m};\mathbf{i}) \propto D(\mathbf{m})(\mathbf{m}\cdot\mathbf{n})$ 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).

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

 // Sample a point on the disk.
 float radius = sqrt(u1);
 float phi = 2.0 * M_PI * u2;
 float x = radius * cos(phi);
 float t = radius * sin(phi);
 float y = lerp(sqrt(1.0 - x * x), t, s);

 // Build an orthonormal basis.
 // The disk is perpendicular to [0, 0, 1].
 float lensq = d.x * d.x + d.y * d.y;
 float2 axisY = lensq != 0.0 ? d / sqrt(lensq) : float2(0.0, 1.0);
 float2 axisX = {axisY.y, -axisY.x};

 // Project the sample point onto a sphere.
 float z = sqrt(max(1.0 - x * x - y * y, 0.0));
 float3 n = float3(axisX * x + axisY * y, z);

 // Unstretch and normalize the microfacet normal.
 return normalize(float3(n.xy * roughness, n.z));
}
```

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

```
float3 SampleUnmaskedNormal(float u1, float u2, float2 roughness,
     float3 dir) {
 // Stretch and normalize the incident direction.
 float3 axisZ = normalize(float3(dir.xy * roughness, dir.z));

 // Compute the limit angle of the sampling range: psi = acot(3cot(
      theta)).
 float sinTheta = length(axisZ.xy);
 float psi = atan(sinTheta / (3.0 * axisZ.z));

 // cos(theta - psi) = cos(theta) * cos(psi) + sin(theta) * sin(psi).
 float limit = axisZ.z * cos(psi) + sinTheta * sin(psi);

 // Sample a point on the disk.
 float radius = sqrt(u1);
 float phi = 2.0 * M_PI * u2;
 float x = radius * cos(phi);
 float t = radius * sin(phi);
 float s = (1.0 + limit) / 2.0;
 float y = lerp(sqrt(1.0 - x * x), t, s);

 // Build an orthonormal basis.
 // The disk is perpendicular to axisZ.
 float lensq = axisZ.x * axisZ.x + axisZ.y * axisZ.y;
 float3 axisX = lensq != 0.0 ? float3(-axisZ.y, axisZ.x, 0.0) / sqrt(
      lensq) : float3(1.0, 0.0, 0.0);
 float3 axisY = cross(axisZ, axisX);

 // Project the sample point onto a sphere.
 float z = sqrt(max(1.0 - x * x - y * y, 0.0));
 float3 n = axisX * x + axisY * y + axisZ * z;

 // Unstretch and normalize the microfacet normal.
 return normalize(float3(n.xy * roughness, n.z));
}
```

Listing 4: Sampling according to $p(\mathbf{m};\mathbf{i}) \propto D(\mathbf{m})(\mathbf{m}\cdot\mathbf{n})$ 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).

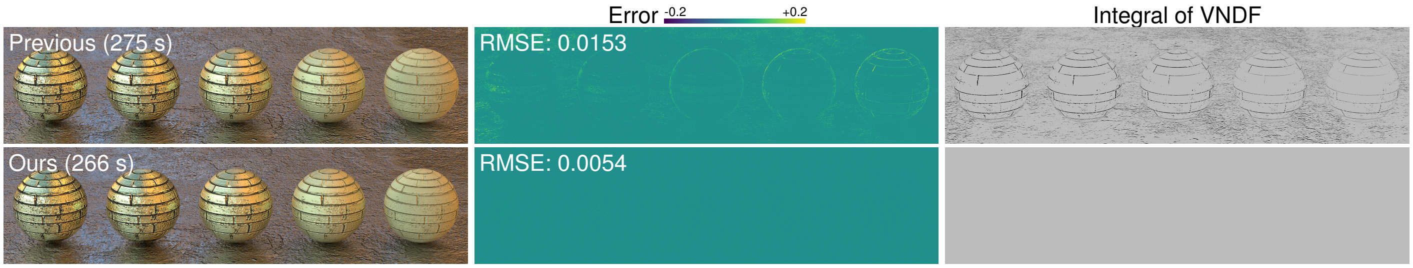
```
float3 SampleNormal(float u1, float u2, float2 roughness, float2 dir,
      float s) {
 // Stretch the incident direction.
 float2 d = dir * roughness;

 // Sample a slope.
 float x = -erfinv(2.0 * u1 * s - 1.0);
 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 ? d / sqrt(lensq) : float2(1.0, 0.0);
 float2 axisY = {-axisX.y, axisX.x};

 // Transform the sampled slope.
 float2 n = axisX * x + axisY * y;

 // Unstretch and normalize the microfacet normal.
 return normalize(float3(n * roughness, 1.0));
}
```

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

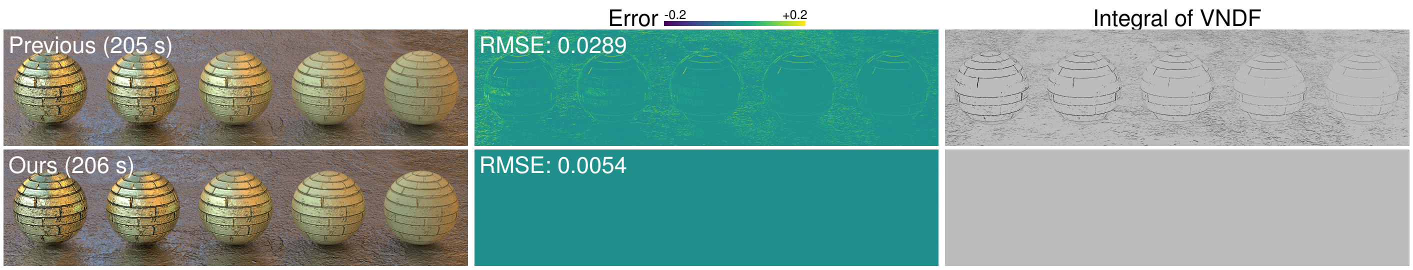
```
float3 SampleUnmaskedNormal(float u1, float u2, float2 roughness,
     float3 dir) {
 // Stretch and normalize the incident direction.
 float3 d = normalize(float3(dir.xy * roughness, dir.z));

 // Compute a lower limit of the sampling range: 3cot(theta).
 float sinTheta = length(d.xy);
 float slopeMin = 3.0 * d.z / sinTheta;

 // Sample a slope.
 float2 p = -SampleVisibleSlope(u1, u2, sinTheta, d.z, slopeMin);

 // Build an orthonormal basis.
 float lensq = d.x * d.x + d.y * d.y;
 float2 axisX = lensq != 0.0 ? d.xy / sqrt(lensq) : float2(1.0, 0.0);
 float2 axisY = {-axisX.y, axisX.x};

 // Transform the sampled slope.
 float2 n = axisX * p.x + axisY * p.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.

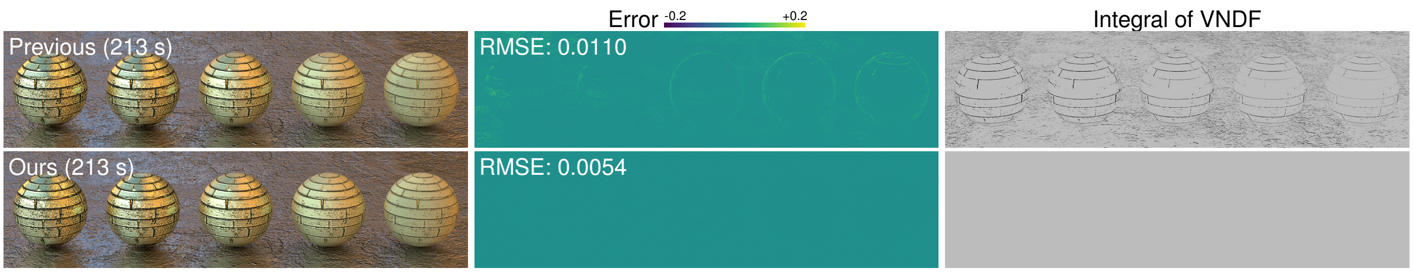
```
float2 SampleVisibleSlope(float u1, float u2, float sinTheta, float
      cosTheta, float slopeMin) {
 // Compute the upper limit cot(theta).
 float slopeMax = cosTheta / sinTheta;

 // Search interval (in the erf() domain).
 float xmin = erf(slopeMin);
 float xmax = erf(slopeMax);

 // Start with a good initial guess based on Mitsuba 2.
 float x = xmax - (1.0 + xmax) * erf(sqrt(-log(max(u1, FLT_MIN))));

 // Normalization and offset for the CDF.
 float SQRT_PI_INV = 0.56418958354775628694807945156077f;
 float tanTheta = sinTheta / cosTheta;
 float cmax = xmax + SQRT_PI_INV * tanTheta * exp(-slopeMax *
     slopeMax);
 float cmin = xmin + SQRT_PI_INV * tanTheta * exp(-slopeMin *
     slopeMin);
 float target = u1 * (cmax - cmin) + cmin;

 // The number of steps is fixed to three for the simplicity.
 for (int i = 0; i < 3; ++i) {
  // Evaluate the CDF and its derivative.
  float erfinvX = erfinv(x);
  float value = x + SQRT_PI_INV * tanTheta * exp(-erfinvX * erfinvX)
      - target;
  float derivative = 1.0 - tanTheta * erfinvX;
  x -= value / derivative;
 }

 float xm = erfinv(x);
 float ym = erfinv(2.0 * u2 - 1.0);

 return float2(xm, ym);
}
```



(a) Smith–Beckmann model



(b) V-cavity–GGX model



(c) V-cavity–Beckmann model

Figure 2: Path tracing using VNDF sampling with previous normalization and our normalization for a normal-mapped scene (2048 samples/pixel, CPU: AMD Ryzen™ 7 3800X). Our normalization avoids a brightening bias caused by the VNDF integral being less than one.

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

[BS63] Petr Beckmann and André Spizzichino. *Scattering of Electromagnetic Waves from Rough Surfaces*. MacMillan, 1963.

[Hd14] Eric Heitz and Eugene d'Eon. Importance sampling microfacet-based BSDFs using the distribution of visible normals. *Comput. Graph. Forum*, 33(4):103–112, 2014.

[Hei14] Eric Heitz. Understanding the masking-shadowing function in microfacet-based BRDFs. *J. Comput. Graph. Tech.*, 3(2):48–107, 2014.

[Hei18] Eric Heitz. Sampling the GGX distribution of visible normals. *J. Comput. Graph. Tech.*, 7(4):1–13, 2018.

[Jak14] Wenzel Jakob. An improved visible normal sampling routine for the Beckmann distribution. Technical report, ETH Zürich, Aug 2014.

[NDVZJ19] Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and Wenzel Jakob. Mitsuba 2: A retargetable forward and inverse renderer. *ACM Trans. Graph.*, 38(6), 2019.

[WMLT07] Bruce Walter, Stephen R. Marschner, Hongsong Li, and Kenneth E. Torrance. Microfacet models for refraction through rough surfaces. In *EGSR '07*, pages 195–206, 2007.