An Improved Visible Normal Sampling Routine for the Beckmann Distribution

Wenzel Jakob

August 2, 2014

Recently, Heitz and D'Eon [2014] proposed a method for importance sampling the distribution of visible normals in the context of microfacet BSDF models. One of their sampling routines internally relies on a discontinuous mapping, which can cause problems in conjunction with Quasi Monte Carlo sampling and Markov Chain Monte Carlo integration. In this report, we develop an alternative method that does not have this drawback.

1 Introduction

Microfacet BSDF models describe the interaction of light with random surfaces composed of microscopic dielectric or conducting facets that are oriented according to a microfacet distribution. Validations against real-world measurements have shown that microfacet models compare favorably against other families of parametric BRDF models, and for this reason they are a popular choice in the context of physically-based rendering.

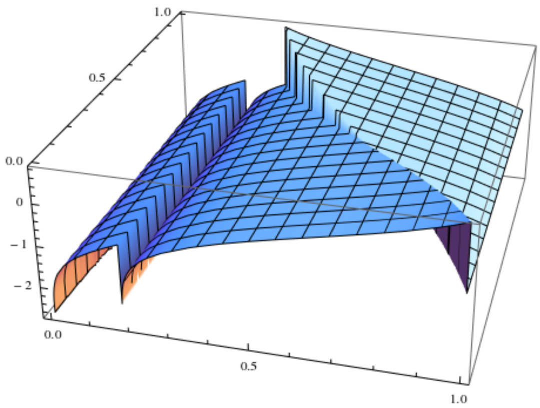
The standard approach [Walter et al. 2007] for importance sampling a microfacet BSDF entails random sampling of a microgeometry normal proportional to a given *microfacet distribution*, followed by the simulation of a reflection or refraction event. However, due to masking effects, only a subset of the microgeometry receives light from any particular direction. The standard technique does not take this into account and generates many samples that contribute little or no energy, which can lead to high variance in renderings.

Recently, Heitz and D'Eon [2014] proposed a method for importance sampling the distribution of *visible* normals, which can lead to significant variance improvements, particularly when dealing with very rough surfaces. However, one technical issue still exists with their approach, specifically for surfaces modeled using the Beckmann microfacet distribution: a key component of their algorithm maps a 2D uniform variate to the x and y slopes of a surface facet. In their implementation, this mapping is *discontinuous*. While not a problem from a theoretical standpoint, this is undesirable for two practical reasons:

- It is common to use stratified samples or Quasi Monte Carlo integration to improve convergence in MC renderings. The effectiveness of these methods is greatly diminished when scattering models use discontinuous mappings in their the sampling routines.

- Some MCMC rendering techniques [Kelemen et al. 2002] expect that small perturbations to the input random variates will result in small changes to sampled directions. This assumption is violated at discontinuities, which can lead to distracting visual artifacts due to variation in the speed of convergence in different parts of the image.

The image below shows a plot of Heitz and D'Eon's mapping from a 2D random variate to x facet slopes for the Beckmann distribution. The mapping was found using a case-by-case analysis of the underlying problem, which gives rise to several "sheets" with discontinuous transitions.



The goal of this report is the design of an alternative continuous mapping for this component.

1.1 Improved Sampling Routine

We only discuss sampling of the x slope (the y slope sampling does not need any modification). The objective of this step is the inversion of the cumulative distribution function

$$C_{\omega_i}(x) := \frac{G_1(\omega_i)\tan(\theta_i)}{2\sqrt{\pi}} \exp\left(-x^2\right) + G_1(\omega_i)\left(\frac{1}{2} + \frac{1}{2}\mathrm{erf}(x)\right)$$

on the interval $[-\infty, \cot\theta_i]$, where G_1 is Smith's shadowing-masking term, ω_i is the incident direction, and θ_i the associated latitude. For further details on this expression, refer to the supplementary material of the paper by Heitz and D'Eon [2014].

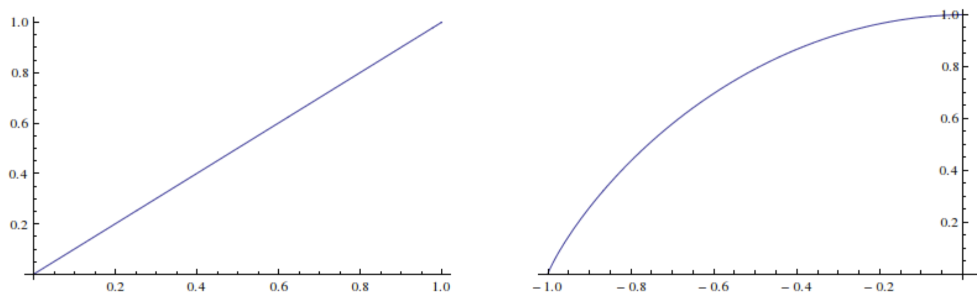
We first discard shared constant factors for simplicity:

$$\hat{C}_{\omega_i}(x) := \frac{\tan(\theta_i)}{\sqrt{\pi}} \exp\left(-x^2\right) + \operatorname{erf}(x) + 1$$

Let us switch to a new set of coordinates x' which relate to x as $x' = \operatorname{erf}(x)$. In these coordinates, the above function is given by

$$\hat{C}_{\omega_i}(\mathrm{x}') = \frac{\tan(\theta_i)}{\sqrt{\pi}} \exp\left(-\operatorname{erf}^{-1}(x')^2\right) + x' + 1$$

This reparameterization leads to a well-behaved function defined on a finite interval $x' \in [-1, \operatorname{erf}(\cot\theta_i)]$. Its shape varies based on the value of $\theta_i \in [0, \pi/2]$. Two (renormalized and scaled) plots are shown below.



At perpendicular incidence, $\hat{C}_{\omega_i}(\mathrm{x}')$ is a linear function (left), and for grazing θ_i it is slightly curved (right). Numerical root-finding techniques based on bracketing intervals can use its monotonicity property to reliably invert $\hat{C}_{\omega_i}(x') = \xi$ to solve for x' given $\xi \in [0, C_{\omega_i}(\operatorname{erf}(\tan\theta_i))]$.

The derivative of this function is also simple to compute and given by

$$\hat{C}'_{\omega_i}(\mathrm{x}') = 1 - \tan(\theta_i) \operatorname{erf}^{-1}(x')$$

Due the close-to-linear nature, Newton-type methods can be expected to converge in a low number of iterations. In Listing 1, we combine both approaches (bracketing intervals and Newton's method) to safely invert $\hat{C}_{\omega_i}$ given θ_i and a uniform variate ξ.

This listing also employs a heuristic that sets the starting point of the root-finding iteration to a point that is close to the correct solution. This fit (obtained in Mathematica) ensures that only 1-2 iterations are required in practice.

```cpp
void sample11(float thetaI, float U1, float U2, float &slope_x, float &slope_y) {
    const float SQRT_PI_INV = 1 / std::sqrt(M_PI);

    float tanThetaI = std::tan(thetaI);
    float cotThetaI = 1 / tanThetaI;

    /* Search interval (in the erf() domain) */
    float a = -1, c = erf(cotThetaI);

    /* Start with a good initial guess (approximation computed in Mathematica) */
    float fit = 1.0f + thetaI * (-0.876f + thetaI * (0.4265f - thetaI * 0.0594f));
    float b = c - (1.0f+c) * std::pow(1.0f - U1, fit);

    /* Normalization factor for the CDF */
    float normalization = 1.0f / (1.0f + c +
        SQRT_PI_INV * tanThetaI * std::exp(-cotThetaI*cotThetaI));

    while (true) {
        /* Bisection criterion -- the oddly-looking boolean expression
           are intentional to check for NaNs at little additional cost */
        if (!(b >= a && b <= c))
            b = 0.5f * (a + c);

        /* Evaluate the CDF and its derivative (i.e. the density function) */
        float invErf = erfinv(b);
        float value = normalization * (1.0f + b +
            SQRT_PI_INV * tanThetaI * std::exp(-invErf*invErf)) - U1;

        if (std::abs(value) < 1e-5f)
            break;

        /* Update bisection intervals */
        if (value > 0.0f)
            c = b;
        else
            a = b;

        /* Perform a Newton step */
        float derivative = (1 - invErf*tanThetaI) * normalization;
        b -= value / derivative;
    }

    /* Now convert back into a slope value */
    slope_x = erfinv(b);

    /* Y slope sampling works as before */
    slope_y = erfinv(2.0f * U2 - 1.0f);
}
```

Listing 1: A C++ listing of the new sampling routine (single precision)

References

HEITZ, E., AND D'EON, E. 2014. Importance sampling microfacet-based BSDFs using the distribution of visible normals. *Computer Graphics Forum* (June).

KELEMEN, C., SZIRMAY-KALOS, L., ANTAL, G., AND CSONKA, F. 2002. A simple and robust mutation strategy for the Metropolis light transport algorithm. *Computer Graphics Forum 21*, 3, 531–540.

WALTER, B., MARSCHNER, S. R., LI, H., AND TORRANCE, K. E. 2007. Microfacet models for refraction through rough surfaces. In *Proceedings of EGSR '07*, Eurographics Association.