Rendering Glints on High-Resolution Normal-Mapped Specular Surfaces

Pramook Khungurn

August 16, 2015

This document was written as I read "Rendering Glints on High-Resolution Normal-Mapped Specular Surfaces" [Yan et al., 2014].

1 Introduction

- This paper seeks to render glints on surfaces equiped with high-resolution normal maps under sharp lighting.
- In such a situation, most of the energy is concentrated in tiny highlight that take up a miniscule fraction of a pixel. So, standard uniform pixel sampling has a hard time hitting the highlight and so results in high variance.
- The paper introduces the concept of $\mathcal{P}$-NDF, which is the unsimplified normal distribution function of a surface path $\mathcal{P}$ seen though a single pixel.
- To render a pixel with a $\mathcal{P}$-NDF, one elementary approach is to repeatedly choose a point on the patch, take its normals, perturb the normal by intrinsic roughness, and add the normal to a bin. After we have done this enough time, the histogram should be representative of the $\mathcal{P}$-NDF, and we can perform normal integral over the bins.

  The problem with the above algorithm is that there is probably only one bin the histogram that contributes any energy. This is because, given the light and the eye directions, the normal that contribute to the glint is the half vector between these two directions. As a result, we wasted resource trying to gather information about a single bin.
- The paper's approach is to evaluate the density of a single normal with a deterministic approach (i.e., analytical integration).

2 Preliminaries

- **Pixel footprint.** The pixel footprint in this paper is not a triangle over the normal map as would normally be used. Instead, the paper uses a Gaussian pixel reconstruction filter on the *uv*-parameterization of the normal map instead.

  The Gaussian does not have to be circular. The covariance matrix can be easily computed by tracing ray differentials.
- **Projected hemisphere.** A normal, which is a hemispherical vector, is represented with a two dimensional vector in the unit disk $\mathcal{D}$. A point $\mathbf{s} = (s,t) \in \mathcal{D}$ represents the unit 3-vector $(s, t, \sqrt{1 - s^2 - t^2})$.

  The *extended unit disk* is the union of $\mathcal{D}$ and the symbol $\perp$, which denotes an invalid normal. This is useful when dealing with normal maps that has invalid normals in it.

- **Normal map.** A normal map is defined as a function $n : \mathbb{R}^2 \to \mathcal{D}$ from points $\mathbf{u} = (u, v)$ in the texture space to normals $\mathbf{s} = (s, t)$.

  The Jacobian of $n(\mathbf{u})$ is denoted $J(\mathbf{u})$. This quantity shows up as change-of-variable factor when we integrate the normal map. In particular, a singular occurs when $\det J(\mathbf{u}) = 0$.

- **Intrinsic roughness.** To avoid singularity caused by $\det J(\mathbf{u}) = 0$, the paper convolves the normal map with a small Gaussian kernel, which introduces intrinsic roughness to the normal map.

- **NDFs.** An NDF is a probability density function on the extended unit disk, with the obvious measure. (I have no idea what this "obvious" measure is. Is it the area measure?)

  This formulation allows the paper to process the NDF very easily. If they want to convolve it with a filter, they just convolve it in 2D.

- **$\mathcal{P}$-NDF.** The $\mathcal{P}$-NDF is the probability density of sampling the footprint $\mathcal{P}$, evaluating the normal, and purturbing by the intrinsic roughness kernel.

- Once we have the $\mathcal{P}$-NDF $D(\mathbf{s})$, we can use it to render specular surface, modeled by a microfacet distribution, as follows. First, we need to determine the half vector $\mathbf{h} = (\mathbf{i} + \mathbf{o})/\|\mathbf{i} + \mathbf{o}\|$ where $\mathbf{i}$ is the light vector and $\mathbf{o}$ is the eye vector. Next, we can plug this into the microfacet BRDF evaluation:

  $$f_r(\mathbf{i}, \mathbf{o}) = \frac{F(\mathbf{i} \cdot \mathbf{n}) G(\mathbf{i} \cdot \mathbf{h}) D(\mathbf{h})}{4(\mathbf{i} \cdot \mathbf{n})(\mathbf{o} \cdot \mathbf{n})}$$

  where $\mathbf{n}$ is the shading frame normal.

3 $\mathcal{P}$-NDF evaluation in flatland and 3D

- To derive at the algorithm the paper uses, it is instructive to work in a simpler flatland situation.
- In the flat land, a normal is specified by a single number in $s \in (-1, 1)$. The full normal vector is given by $(s, \sqrt{1 - s^2})$.
- The normal map can be written as a function $n(u)$.
- The pixel reconstruction filter is the Gaussian probability distribution $G_p(u)$.
- The $\mathcal{P}$-NDF is the probability density of $n(u)$ according to the following process:
  - Pick u according to $G_p(u)$.
  - Evaluate $n(u)$.

  As such, for a value $s \in (-1, 1)$, we have that the $\mathcal{P}$-NDF $D(s)$ is given by:

  $$D(s) = \int_{-\infty}^{\infty} G_p(u) \delta(n(u) - s)\ \mathrm{d}u.$$

  To evaluate this integral, we perform the change of variable with $v(u) = n(u) - s$. As a result,

  $$\mathrm{d}u = \frac{\mathrm{d}v}{|n'(u)|}.$$

  Therefore,

  $$D(s) = \int_{v(-\infty)}^{v(\infty)} G_p(u) \frac{\delta(v)}{|n'(u)|}\ \mathrm{d}v = \sum_i \frac{G(u_i)}{|n'(u_i)|}$$

  where u_i is such that $n(u_i) = s$.

- The above formulation has problems when $n'(u_i) = 0$. To avoid this problem, the paper replaces the Direct delta function with a small Gaussian $G_r(s)$:

$$D(s) = \int_{-\infty}^{\infty} G_p(u) G_r(n(u) - s)\ \mathrm{d}u.$$

  Momentarily, let us substitute $n(u)$ with v. As a result, we have that

$$G_p(u) G_r(n(u) - s) = G_p(u) G_r(v - s).$$

  Notice that $G_p(u)G_r(v-s)$ is a 2D Gaussian, which we will replace by the symbol $G_c[\mathcal{P}, s](u, v)$. In other words,

$$G_p(u) G_r(n(u) - s) = G_c[\mathcal{P}, s](u, n(u)).$$

  As such,

$$D(s) = \int_{-\infty}^{\infty} G_c[\mathcal{P}, s](u, n(u))\ \mathrm{d}u.$$

  A way to visualize the situation is that we travel along the graph $(u, n(u))$, evaluate the 2D Gaussian on each point of the graph, and integrate the value with the du measure.

- Now, if we move to 3D (reduced to 2D vector in $\mathcal{D}$), the situation is the same. The only differences are that the variables are now 2D vectors:

$$D(\mathbf{s}) = \int_{\mathbb{R}^2} G_c[\mathcal{P}, s](\mathbf{u}, n(\mathbf{u}))\ \mathrm{d}\mathbf{u}.$$

  Here, $G_c[\mathcal{P}, \mathbf{s}]$ is a 4D Gaussian. Also, instead of integrating the values of the Gaussian on the graph $(u, n(u))$, we integrate on the 2D surface $(\mathbf{u}, n(\mathbf{u}))$ embeded in 4D space.

4 Actually performing the integration

- The normal map is discretized into triangles (why not squares?). The normals are linearly interpolated across the triangles. The paper divides each pixel of the normal map into 4×4 subpixels. Each square pixel is covered by 2 triangles. So, there are 32 triangles per pixel.

- The goal then becomes to integrate the 4D Gaussian values over the each triangles:

$$I = \int_{\Delta} G_c(\mathbf{u}, n(\mathbf{u}))\ \mathrm{d}\mathbf{u} = \int_{\Delta} G(\mathbf{u})\ \mathrm{d}\mathbf{u}.$$

- Suppose the triangle Δ has vertices at (u_0, v_0), (u_1, v_0), and (u_0, v_1). Then,

$$I = \int_{u_0}^{u_1} \left(\int_{v_0}^{f(u)} G(u, v)\ \mathrm{d}v \right) \mathrm{d}u$$

  where

$$f(u) = \frac{(u_1 - u)v_1 + (u - u_0)v_0}{u_1 - u_0}.$$

- Let us expand the G term a little bit. We have that

$$G(u, v) = G_p(u, v) G_r(n(u, v) - (s_x, s_y))$$

References

[Yan et al., 2014] 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 Trans. Graph.*, 33(4):116:1–116:9.