

Supplementary Material: Scratch-based Reflection Art via Differentiable Rendering

PENGFEI SHEN*, University of Science and Technology of China, China

RUIZENG LI*, University of Science and Technology of China, China

BEIBEI WANG†, Nankai University and Nanjing University of Science and Technology, China

LIGANG LIU†, University of Science and Technology of China, China

1 IMPLEMENTATION DETAILS

To evaluate our scratch BRDF, we first find the intersecting scratches for each pixel with OptiX and then compute the NDF contribution of each scratch (Eqn. (10) (main paper)) in Dr.JIT. In this section, we provide the details of these two steps.

1.1 Ray-scratch intersection

To find the scratches which intersect with a pixel's footprint, we shoot a ray from each footprint's center and search for the scratches by traversing the BVH. Since the physical width of scratches might be much smaller than a pixel's footprint, shooting one ray only from the center fails to find all the scratches. One way is to shoot more rays randomly in the pixel, which increases the time cost and introduces variance. Therefore, we propose a simple way to overcome the missing scratch issue by setting the scratches' width as the maximum radius of the footprint. For our optimization purpose, this could be acquired only once in the first iteration, after collecting the footprints buffer.

1.2 Unified computation for different intersection types

For all the scratches which intersect with the current pixel's footprint, we accumulate their NDF contribution. The NDF of each scratch is integral of the point NDF over the scratch area, which has been transformed into the integral on the boundaries, using Green's theorem [Riley et al. 2006], as discussed in Section 3.3 (main paper).

There are eight types of intersections between a scratch and a footprint, as shown in Figure 1. Handling these types with different branches will be highly inefficient on GPU. To avoid this issue, we use a simple but universal way. Each footprint is a quadrilateral with four edges, which indicates that there are at most four forms

*Joint first authors.

†Corresponding authors.

Authors' addresses: Pengfei Shen, University of Science and Technology of China, China, jerry_shen@mail.ustc.edu.cn; Ruizeng Li, University of Science and Technology of China, China, pb17061297@mail.ustc.edu.cn; Beibei Wang, Nankai University and Nanjing University of Science and Technology, China, beibei.wang@njust.edu.cn; Ligang Liu, University of Science and Technology of China, China, lglu@ustc.edu.cn.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM. 0730-0301/2023/8-ART \$15.00 https://doi.org/10.1145/nnnnnnnn.nnnnnnnn

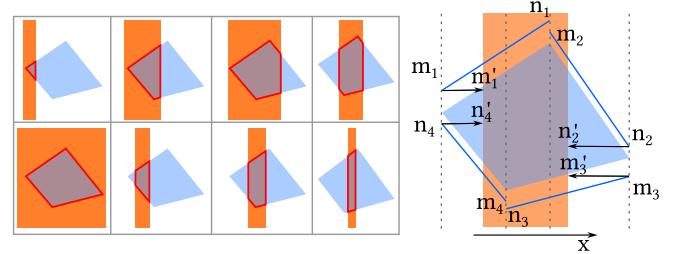


Fig. 1. Eight types of intersections between a footprint (blue) and a scratch (orange). The integral boundaries are marked in red.

of $S(x)$ in Eqn. (10) (main paper). We denote the x -coordinate of j -th edge's endpoints as m_j, n_j :

$$\begin{aligned} m'_j &= \min(w, \max(-w, m_j)) \\ n'_j &= \min(w, \max(-w, n_j)). \end{aligned} \quad (1)$$

Note that the parameter w is the physical width of the dent, rather than the width \tilde{w} defined in the main paper. w is solved by evaluating the constraint, $h(w; \tilde{w}, d) = 0$,

$$-\frac{1}{2}\tilde{w} \log\left(1 - \frac{4w^2}{\tilde{w}^2}\right) - d = 0. \quad (2)$$

Then the integral can be rewritten as

$$\hat{D}_i(\mathbf{h}) = \sum_{j=1}^{j=4} \int_{m'_j}^{n'_j} S(x) D_x(\mathbf{h}, \mathbf{n}(x)) dx, \quad (3)$$

In this way, all different cases can be handled universally. This is desired by GPU computing as well as auto differential libraries (in our case, Dr.JIT). Note that the minimum and maximum operations do not harm the ability to perform the auto differential.

2 CLOSED-FORM INTEGRATION OF SCRATCH NDF

It is recommended for the readers to calculate the full formula of the closed-form integral with symbolic computing software (e.g., Mathematica) and export C++ code. For completeness, we provide some of the details here.

Beginning from the Eqn. (10) (main paper),

$$\hat{D}_i(\mathbf{h}) = \sum_{k=1}^{K=K} \int_{p_x^k}^{p_x^{k+1}} S(x) D_x(\mathbf{h}, \mathbf{n}(x)) dx, \quad (4)$$

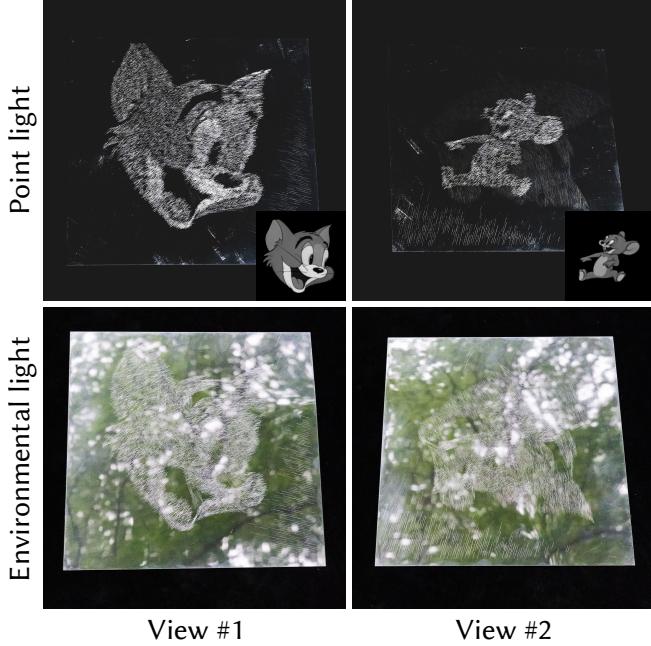


Fig. 2. In the second row, we show our board with a less controlled light setting. The photos are taken under sunlight with surroundings. Comparing to the first row, where the view and light are strictly set to the optimization settings, the images are less visible.

where $S(x) = a_k x + b_k$. We expect to find the indefinite integral $I(x)$, such that

$$I(p_x^{k+1}) - I(p_x^k) = \int_{p_x^k}^{p_x^{k+1}} S(x) D_x(\mathbf{h}, \mathbf{n}(x)) dx \quad (5)$$

It is required to substitute the detailed expression of $D_x(\mathbf{h}, \mathbf{n}(x))$ into Eqn. (4).

$$D_x(\mathbf{h}, \mathbf{n}(x)) = \frac{\alpha^2}{\pi \left(\frac{(\alpha^2-1)(\mathbf{h}_z(\tilde{w}^2-4x^2)-4\mathbf{h}_x\tilde{w}x)^2}{(\tilde{w}^2+4x^2)^2} + 1 \right)^2}. \quad (6)$$

Let $r^2 = 1 - \alpha^2$ and simplify it:

$$\begin{aligned} S(x) D_x(\mathbf{h}, \mathbf{n}(x)) = & - \left(r^2 - 1 \right) \left(\tilde{w}^2 + 4x^2 \right)^4 (a_k x + b_k) / \pi / \\ & \left(-4\mathbf{h}_x r \tilde{w} x + \tilde{w}^2 (\mathbf{h}_z r + 1) - 4x^2 (\mathbf{h}_z r - 1) \right)^2 / \\ & \left(-4\mathbf{h}_x r \tilde{w} x + \tilde{w}^2 (\mathbf{h}_z r - 1) - 4x^2 (\mathbf{h}_z r + 1) \right)^2. \end{aligned} \quad (7)$$

Evaluating the closed-form integral takes two main steps. First, for the two equations

$$\begin{aligned} -4\mathbf{h}_x r \tilde{w} z + \tilde{w}^2 (\mathbf{h}_z r + 1) - 4z^2 (\mathbf{h}_z r - 1) &= 0, \\ -4\mathbf{h}_x r \tilde{w} z + \tilde{w}^2 (\mathbf{h}_z r - 1) - 4z^2 (\mathbf{h}_z r + 1) &= 0, \end{aligned} \quad (8)$$

we can find four complex roots, $\{z_1, z_2, z_3, z_4\}$. The final form of $I(x)$ is as followed.

$$I(x) = \frac{\alpha^2}{16\pi (\mathbf{h}_z^2 r^2 - 1)^4} \left(u(x) + \frac{r^2 \tilde{w} (\mathbf{h}_z^2 r^2 - 1)}{\mathbf{h}_x^2 r^2 + \mathbf{h}_z^2 r^2 - 1} \sum_{i=1}^4 v(x; z_i) \right). \quad (9)$$

$u(x)$ and $v(x; z)$ are two locally defined functions, representing two yet quite complex terms in $I(x)$. They can be calculated and directly exported by symbolic calculation software. Here we omit the result. There are a lot of repeating terms in $u(x)$ and $v(x; z)$. In C++ implementation, using temporary variables can increase performance significantly.

3 RESULTS UNDER COMPLEX LIGHTING SETUP

In Figure 2, we show an example of our board under a more complex lighting setup. The photos are taken under sunlight with environmental illumination. Since our scratch-based reflection art is optimized under a single point light, as expected, the images are less visible due to the non-optimal light direction and the specular reflection of the surroundings.