Micro-specular-highlight real-time rendering

Namo Podee[1,a)] Nelson Max[2] Kei Iwasaki[3] Yoshinori Dobashi[1]

Abstract: Specular highlights are a critical factor for rendering high-quality and realistic images. However, consistent rendering of specular highlight smaller than a pixel size is a challenging problem. The problem usually arises when the lighting is evaluated at only the center of a pixel. An intense highlight can occur in a small area that is not at the center of its pixel. We propose a method that renders in real-time a sub-pixel specular-highlight. Our approach finds a reflection vector for each pixel that creates the brightest highlight by interpolating between four reflection vectors at the pixel corners. Then we compute the lighting with the reflection vector and multiply the result by an appropriate weight to account for the fraction of the pixel area that reflects the light source.

Keywords: real-time rendering, anti-aliasing, specular reflection, specular highlight

1. Introduction

A specular highlight or a specular reflection shows an interaction between a surface and a light source, which renders a characteristic of the material of the surface. Thus, specular highlights are crucial for the quality of a rendering result. However, the conventional method still has trouble with rendering small or complicated sharp specular highlights in real-time.

A small specular highlight is created by a small light source and a mirror-like surface. The problem arises when the highlight itself is too tiny to be captured consistently. The conventional rasterization method usually samples once per pixel at the center, and if the highlight is smaller than the pixel, then the process cannot consistently find the highlight. This inconsistency creates aliasing in the final result.

We can increase the number of samples per pixel to remove the aliasing, but it is computationally expensive; it requires a large number of samples, particularly for mirror-like surfaces. Another solution is to find the contribution from the whole surface within the pixel by approximating the surface's normal variation as a normal distribution function, which is usually represented by a smooth function. However, this approach usually results in a blurred highlight; it cannot render the distinct appearance of the surface with a sharp specular highlight.

The use of normal mapping and deferred shading also escalates the problem. Normal mapping adds more detail and complexity to the surface, while deferred shading expands the number of possible light sources, which potentially introduces more aliasing.

1 Hokkaido University
2 University of California, Davis
3 Wakayama University
a) namo.podee@gmail.com

We propose a method that analytically chooses a reflection vector that produces the most intense reflection within a pixel for each light source. Then our method weights the result with the ratio of the light source size within the pixel reflection space. Our method produces in real-time a consistent specular reflection without aliasing.

Our main contributions are:

- An approximation of a reflection vector variation within a pixel as a linear combination of four reflection vectors at the four corners of the pixel.
- An analytical method for finding the reflection vector producing the strongest reflected intensity within a pixel.
- A weighting of the lighting of the strongest reflection vector.

2. Related work

Many previous methods focus on the rendering of the specular highlight. However, not much research focuses on an accurate and sharp highlight in real-time. We categorize the related work into three groups: smooth function approximation, multi-sampling, and procedural generation.

Smooth function approximation approximates a normal distribution function (NDF) within a pixel with a certain smooth function. LEAN mapping [5] approximates and precomputes an NDF of a normal map with a Gaussian function. Then LEADR mapping [1] added shadowing and a physically-based term into the LEAN mapping for better quality. The frequency domain normal map [2] uses spherical harmonics for low-frequency data and a spherical von MisesFisher distributions for high-frequency data. The three methods can remove the aliasing in real-time, but their reflection results are still not sharp enough and lack details.

Multi-sampling uses multiple samples to obtain the ac-

© 2021 Information Processing Society of Japan

curate result efficiently. Some methods for temporal anti-aliasing [8] reuse the previous frame's samples to increase the efficiency. However, the information of a sample in the current frame might not be in the predicted location or not available at all in the previous frame. This causes the method to fall back to fewer samples, increasing the aliasing. Yan et al. [6], [7] introduce a technique that accurately renders glints by sampling a whole normal map within a pixel footprint with a computationally efficient strategy. Zeltner et al. [9] sample light paths related to caustics and glints efficiently by using manifold walking, which helps light paths to converge to important light paths. Loubet et al. [4] importance-sample caustic and glint light paths by approximating normal variation as a linear interpolation between three normal vectors at vertices of a triangle. This work uses an assumption similar to our work, which will be explained later. All the methods in this group except for the temporal anti-aliasing are computationally too expensive to be computed in real-time.

Procedural generation creates a surface model, which can simplify the computation. Jakob et al. [3] proposed a stochastic model of randomized metallic flakes, which was used to evaluate a surface in each pixel to render glints. However, the evaluation is still too expensive to be done in real-time. Zirr and Kaplanyan [10] proposed a procedural model of micro-detail that produced a glittery effect. The procedural model is consistent due to stable multi-scale noise generation, and runs in real-time. However, both methods support only limited types of surface models.

3. Proposed method

Our main objective is to render a highlight of a light source that is smaller than a pixel by considering the whole perfect specular surface within a pixel to capture the small highlight. We will introduce a brief overview of our method first, then move on to the detail of each component of the method.

The basic idea is to find the sample position that produces the brightest highlight instead of the center within a pixel to compute the intensity. Then we multiply the intensity with the weight representing a fraction of the light source area computed with the pixel reflection space. These two components are called *reflection vector selection* and *result weighting*. **Fig. 1** shows the algorithm of our method.

3.1 Reflection vector selection

A fraction of the surface within a pixel is in general a curved surface patch. Our method first samples reflection vectors at the corners of the pixel, as shown in **Fig. 2**. We assume that a reflection vector within the pixel can be represented as a linear combination of these four vectors. Let us consider a light plane defined as a plane located at $p+l$ with a normal vector of l, where p is the surface center and l is the unit vector toward the light direction. Then, we compute a quadrilateral by projecting the four reflection vectors onto the light plane as shown in **Fig. 3**. Note that we assume the

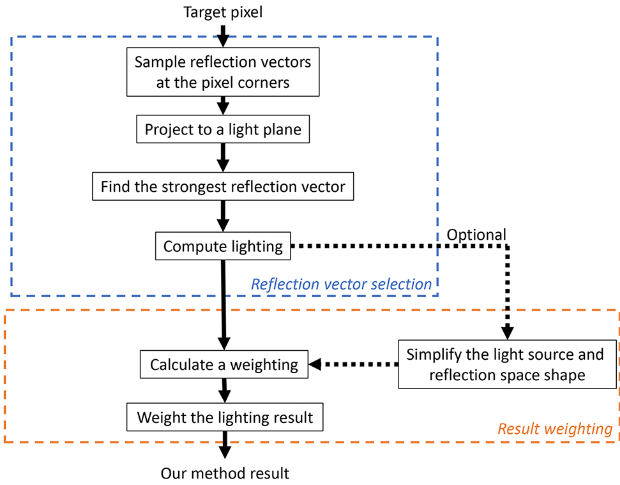


Fig. 1: An overview of our method.

refection vectors originate from the center of the surface. This quadrilateral defines the *reflection space*, that is, any reflection vectors within the pixel are projected within this quadrilateral. We assume a reflection vector within a pixel is obtained by bilinear interpolation of the four vectors.

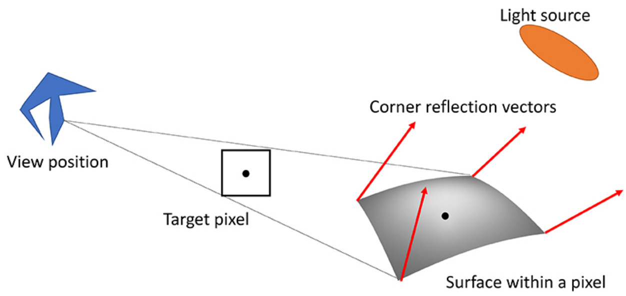


Fig. 2: A curved surface patch inside a pixel and four samples of reflection vectors at the corners.

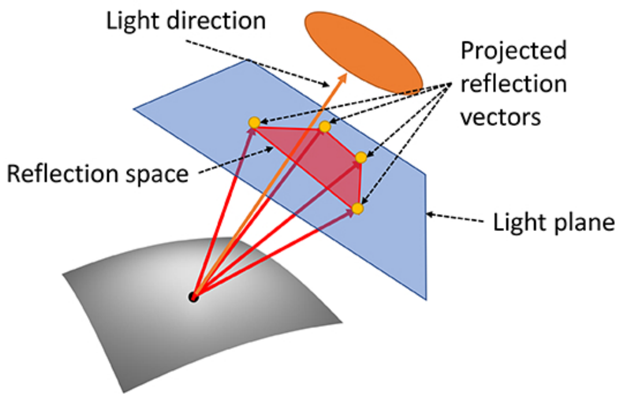


Fig. 3: Definitions of the light plane and the reflection space.

We then compute the most intense reflection vector by finding the point inside the reflection space closest to the center of the projected light source with this simple assumption.

© 2021 Information Processing Society of Japan

The closest point is analytically determined by using geometric analysis given the shape of the light source. Currently, our implementation supports only a disk light source. The closest point to the disk light source can be solved by checking the projected position of the the center of the light source. If the center is in the reflection space, the closest point is the center. Otherwise, the closest point is the point on the reflection space edge with the closest distance to the center.

Then we use the direction from the surface center to the closest point as the most intense reflection vector to compute the lighting. The result is the brightest highlight of any point within the pixel.

3.2 Result weighting

To compute the correct reflection intensity for a pixel, we need to compute an integral of the reflected intensity over a surface within the pixel. However, our method has only a single sample of the brightest reflection. We approximate the intensity with the product of the brightest reflection with the fraction of the projected area of the light source within the reflection space. This approximation is valid when the surface exhibits mirror-like reflections and is illuminated by a uniformly bright light source.

We can calculate the fraction of the surface area in the reflection space on the light plane due to our assumption that the reflection of the surface is bilinearly interpolated, and an additional assumption that the Jacobian determinant of the bilinear mapping is constant (which is not always the case). Then area on the pixel is proportional to area on the reflection space. So we only need to find the area of the intersection between the projected light source and the reflection space and the area of the reflection space. Both areas are solved by using geometric analysis on the 2D light plane.

However, the intersection calculation might be too expensive computationally. Hence we further approximate the areas of the projected light source and/or the shape of the reflection space. We are currently using two approximation types: we approximate the projected light source by a circle, and the reflection space with a quad.

4. Results

We implemented our method using OpenGL on a laptop with an Intel Core i7 @ 2.50Ghz, 16 GB Memory, and an NVIDIA GeForce GTX 860M. We only experiment with a disk light source at the moment. **Fig. 4** shows the comparison between the reference image, the conventional method result, LEAN mapping method, and our method result. The reference image is calculated by the tiling 64x supersampling method, which separates a target result into multiple tiles and renders each of them sequentially. **Table 1** shows the root mean square error (RMSE) of these methods.

Our method shows an improvement in both the quality and stability of a specular highlight over the conventional method and LEAN mapping. LEAN mapping overly blurs the reflection and does not work at close distances. However, our method fails to capture the reflection at far distances because there is high normal variation within a pixel, so our curved surface approximation with four reflection vectors is not accurate enough.

Table 1: The RMSE comparison.

Rendering Method	RMSE
Conventional	0.0845
LEAN mapping	0.0977
Our Method	0.0826

5. Conclusion

Our method can consistently capture a small highlight on a detailed reflective surface without aliasing. However, it doesn't work well when there is a lot of detail inside one pixel. This is still a work in progress. We need to check the error from our assumptions and conduct experiments with different light sources and surface types. We also need to compare our method with the other state of the art techniques, which are temporal anti-aliasing, LEADR mapping, and frequency domain normal mapping.

Acknowledgment

This work was supported by JSPS KAKENHI Grants Number JP15H05924, JP18H03348, and JP20H05954.

References

[1] Dupuy, J., Heitz, E., Iehl, J.-C., Poulin, P., Neyret, F. and Ostromoukhov, V.: Linear Efficient Antialiased Displacement and Reflectance Mapping, *ACM Trans. Graph.*, Vol. 32, No. 6, pp. 211:1–211:11 (online), DOI: 10.1145/2508363.2508422 (2013).

[2] Han, C., Sun, B., Ramamoorthi, R. and Grinspun, E.: Frequency Domain Normal Map Filtering, *ACM SIGGRAPH 2007 Papers*, SIGGRAPH '07, New York, NY, USA, Association for Computing Machinery, p. 28–es (online), DOI: 10.1145/1275808.1276412 (2007).

[3] Jakob, W., Hašan, M., Yan, L.-Q., Lawrence, J., Ramamoorthi, R. and Marschner, S.: Discrete Stochastic Microfacet Models, *ACM Trans. Graph.*, Vol. 33, No. 4 (online), DOI: 10.1145/2601097.2601186 (2014).

[4] Loubet, G., Zeltner, T., Holzschuch, N. and Jakob, W.: Slope-Space Integrals for Specular next Event Estimation, *ACM Trans. Graph.*, Vol. 39, No. 6 (online), DOI: 10.1145/3414685.3417811 (2020).

[5] Olano, M. and Baker, D.: LEAN Mapping, *Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '10, New York, NY, USA, ACM, pp. 181–188 (online), DOI: 10.1145/1730804.1730834 (2010).

[6] Yan, L.-Q., Hašan, M., Jakob, W., Lawrence, J., Marschner, S. and Ramamoorthi, R.: Rendering Glints on High-Resolution Normal-Mapped Specular Surfaces, *ACM Trans. Graph.*, Vol. 33, No. 4 (online), DOI: 10.1145/2601097.2601155 (2014).

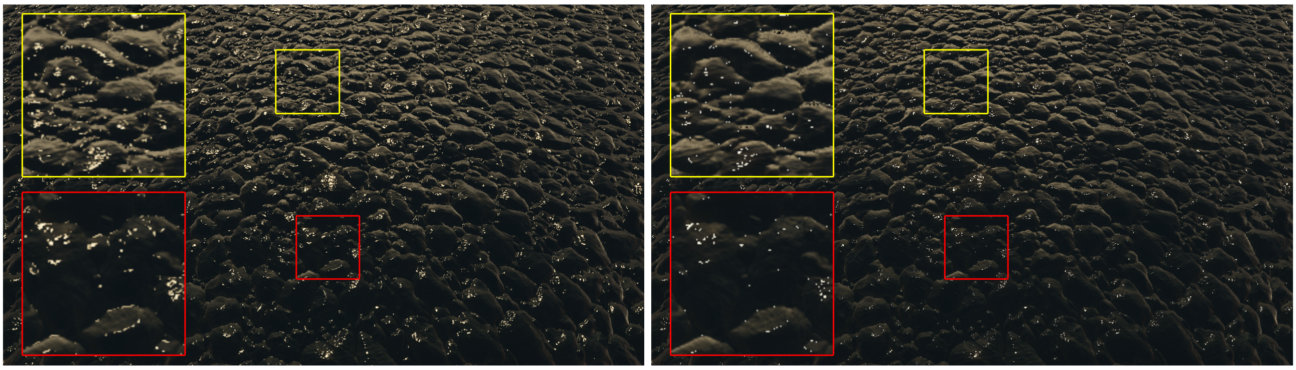
[7] Yan, L.-Q., Hašan, M., Marschner, S. and Ramamoorthi, R.: Position-Normal Distributions for Efficient Rendering of Specular Microstructure, *ACM Trans. Graph.*, Vol. 35, No. 4 (online), DOI: 10.1145/2897824.2925915 (2016).

[8] Yang, L., Liu, S. and Salvi, M.: A Survey of Temporal Antialiasing Techniques, *Computer Graphics Forum*, Vol. 39, No. 2, pp. 607–621 (online), DOI: https://doi.org/10.1111/cgf.14018 (2020).

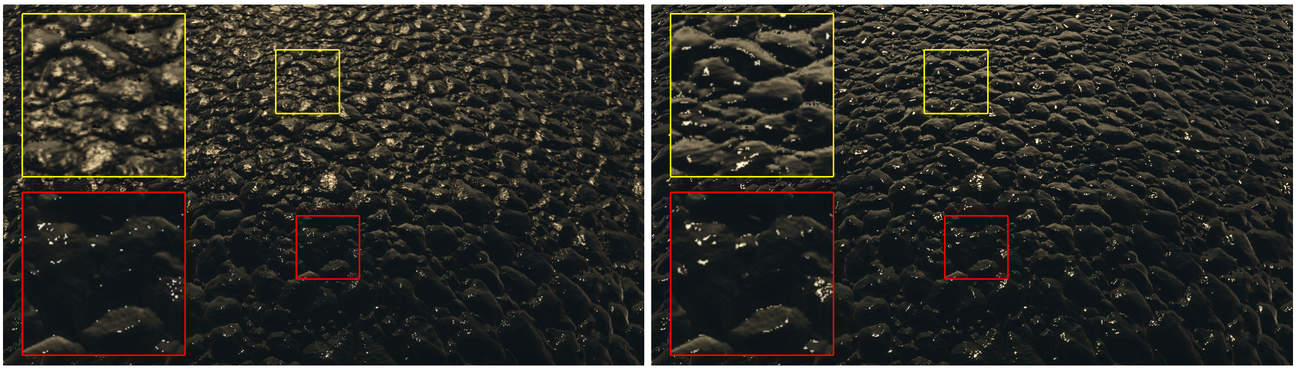
[9] Zeltner, T., Georgiev, I. and Jakob, W.: Specular Manifold Sampling for Rendering High-Frequency Caustics and Glints, *ACM Trans. Graph.*, Vol. 39, No. 4 (online), DOI: 10.1145/3386569.3392408 (2020).

[10] Zirr, T. and Kaplanyan, A. S.: Real-Time Rendering of Procedural Multiscale Materials, *Proceedings of the 20th*

ⓒ 2021 Information Processing Society of Japan



(a) Reference image (172 ms) (b) Conventional (21 ms)



(c) LEAN mapping (27 ms) (d) Our method (31 ms)

Fig. 4: The comparison result with a gravel normal map and a disk light source.

ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games, I3D '16, New York, NY, USA, Association for Computing Machinery, p. 139–148 (online), DOI: 10.1145/2856400.2856409 (2016).

© 2021 Information Processing Society of Japan