Efficient Specular Glints Rendering with Differentiable Regularization

Jiahui Fan[†], Beibei Wang[†‡], Wenshi Wu, Miloš Hašan, Jian Yang[‡] and Ling-Qi Yan

Abstract—Rendering glinty details from specular microstructure enhances the level of realism in computer graphics. However, naive sampling fails to render such effects, due to insufficient sampling of the contributing normals on the surface patch visible through a pixel. Other approaches resort to searching for the relevant normals in more explicit ways, but they rely on special acceleration structures, leading to increased storage costs and complexity. In this paper, we propose to render specular glints through a different method: differentiable regularization. Our method includes two steps: first, we use differentiable path tracing to render a scene with a larger light size and/or rougher surfaces and record the gradients with respect to light size and roughness. Next, we use the result for the larger light size and rougher surfaces, together with their gradients, to predict the target value for the required light size and roughness by extrapolation. In the end, we get significantly reduced noise compared to rendering the scene directly. Our results are close to the reference, which uses many more samples per pixel, although our method cannot guarantee unbiased convergence to the reference. The overhead for differentiable rendering and prediction is small, so our improvement is almost free. We demonstrate our differentiable regularization on several normal maps, all of which benefit from the method.

Index Terms—Microstructure, Glints Rendering, Differentiable Rendering

1 Introduction

Many real materials show complex specular reflections caused by intricate microstructures. Examples include car paints, scratched and brushed metals, plastics, and many more. The effect is most pronounced under sharp illumination, but also becomes most challenging to render under these conditions (Figure 1).

The naive rendering method (sampling pixels uniformly) suffers from obvious noise under sharp light sources. The reason for this inefficient sampling is that the pixel footprint covers an area of the normal map which has a complex normal distribution; if the surface material is highly specular and the light source is small, this makes the light paths difficult to sample (see Figure 1). Hence, a large number of samples is required to get a converged result with a naive approach.

The work of Yan et al. [1] was the first non-trivial solution to this problem, and has resulted in a new level of realism for rendering specular highlights. This method and subsequent work [2] use high-resolution normal maps to explicitly define every microfacet normal. However, these methods rely on acceleration hierarchies for efficient queries, leading to increased complexity and storage.

In this paper, we propose a completely different solution: differentiable regularization. Our observation is that increasing either the surface roughness or the light source size makes the rendering problem easier: it is much less difficult to find paths with significant contributions. Our key idea is to sample these "easy" paths first and then use them to predict the "difficult" result by extrapolation. We first use differentiable path tracing to render the scene with a slightly larger light size and slightly rougher material, while recording the gradient with respect to light size and roughness. Then we predict the result of the target light size and roughness by extrapolation from the rendered result and the two gradients. We also propose a Gaussian light source which is convenient for differentiable rendering.

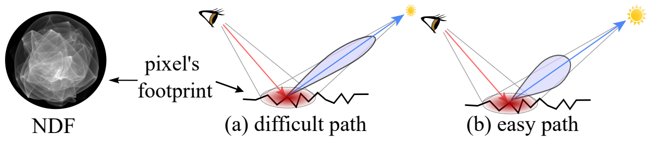


Fig. 1. The key challenge in glints rendering. Each pixel projects onto an area of the high-resolution normal map or height field, which yields a complex normal distribution. If the material of the surface is close to specular and the light is small, it becomes non-trivial to find the normals that give rise to valid light paths. The key idea of this paper is to solve the difficult case by first solving an easier case, where the light source size is larger, and the surface is rougher; then extrapolating to the difficult case using gradients.

Our method is able to extrapolate a result close to the ground truth, with much less noise than rendering naively, but is algorithmically much simpler and cleaner that previous solutions based on acceleration structures. The method works well for a variety of normal maps and also naturally preserves temporal coherence.

Path regularization has been successfully used for caustics and other effects. However, we are not aware of previous work using differentiable regularization, and believe this concept is novel in the rendering space. The main contributions of our method can thus be summarized as:

- The concept of *differentiable regularization*: an approach where some scene parameters are first adjusted to make rendering easier, then extrapolation is used towards the target values of the parameters, based on their gradients.

- [†] *Joint first authors.*
- [‡] *Corresponding authors.*
- *J. Fan, B. Wang, W. Wu and J. Yang are with the school of Computer Science and Engineering, Nanjing University of Science and Technology, China.*
- *M. Hašan is with the Adobe Research.*
- *L.-Q. Yan is with the University of California, Santa Barbara.*

- An application of differentiable regularization to specular glint rendering, using light size and roughness as the regularized parameters.

Our solution is a bias/variance trade-off: it is no longer an unbiased Monte Carlo estimator of the reference, and even with many samples, there will be some remaining bias. However, we will see that this bias does not take the form of simple blurring, as is common with previous bias/variance trade-offs. This is important: for complex specular reflection, a user typically cares more about appropriate perceptual sharpness and overall distribution of glints, rather than unbiased accuracy. The simplicity, performance and temporal stability of our method could make it an attractive alternative for practical applications.

In the next section, we review some of the previous work on glints rendering. In section 3, we present our method. We present our results, compare with previous work and analyze performance in section 4, and conclude in section 5.

2 Related work

We first briefly review previous work on microstructure rendering, and then introduce related work on regularization and differentiable rendering.

2.1 Microstructure rendering

Surface reflectance in computer graphics is typically described using microfacet theory [3], which uses smooth analytic functions such as Beckmann [4] and GGX [5] to model the distribution of surface normals. More recently, Yan et al. [1] introduced the idea of using patch-local normal distribution functions ($\mathcal{P}$-NDFs) to accurately compute the local normal distributions from explicit specular microstructure such as bumps, brushes, scratches and metallic flakes, whose microgeometry is defined using very high-resolution normal maps. Yan et al. [2] proposed a method to accelerate this computation, and later extended the techniques to handle wave optics effects [6]. All these methods share a common problem with storage cost: the microstructures have to be defined at high resolutions (e.g., $1-10$ microns per texel), which either requires very large textures (and associated acceleration structures) or leads to tiling artifacts. Recently, Wang et al. [7] proposed a procedural normal map blending approach allowing dynamic point query and range query, which leads to constant storage cost, but still has significant algorithmic complexity. Kuznetsov et al. [8] presented a Generative Adversarial Network to generate NDFs resembling training data. Zhu et al. [9] also proposed a normal map synthesis approach for glints rendering to solve the storage issue, and improved the performance by clustering the Gaussian elements. Compared to these works, our method does not use any hierarchies nor neural networks, thus avoiding both expensive storage and implementation complexity, and yielding a much simpler solution.

Since explicit microstructure is costly to store, a series of methods were designed to model specific effects without the storage requirement. Jakob et al. [10] introduce a procedural BRDF that produces glitter effects from implicit mirror-like flake distributions without explicitly storing the flakes. Wang et al. [11] proposed a separable model, decoupling the spatial and angular domain, which leads to faster glints evaluation. They also proposed a bi-scale filterable model which avoids the expensive search for very large footprints. Later, Wang et al. [12] proposed a three-scale model for real-time glints rendering with environment maps. However, none of these works are extensible to other kinds of microgeometry, such as brushes and scratches. Chermain et al. [13] also proposed a real-time glints rendering approach, by using dictionary of 1D marginal distributions. Recently, Chermain et al. [14] proposed a real-time geometric glint anti-aliasing approach. These works achieve great improvements in performance and storage for computing the glint effect, but tackle the hard case of small lights and high roughness through improved algorithms and data structures. On the other hand, we explore a very different direction, reformulating the problem as extrapolation from an easier case. The lack of algorithmic complexity in our solution could be an important benefit when considering its addition to already complex rendering systems.

Raymond et al. [15] model surfaces as the mixture of a base surface and a collection of 1D scratches, later extended by Werner et al. [16] for wave optics effects and by Velinov et al. [17] for real-time performance; these methods work well for scratches but do not support other appearances. The method of Zirr and Kaplanyan [18] dynamically adds micro-level details to a predefined macro-scale BRDF, but is focused on real-time performance, not on accurate simulation of the appearance of a specific microgeometry.

Zeltner et al. [19] presented a specular manifold sampling technique, which is able to handle glints, reflective/refractive caustics, and specular-diffuse-specular light transport in a Monte Carlo framework. Chermain et al. [20] introduced an importance sampling scheme to optimally sample the multi-lobe visible normal distribution functions for procedural glittering BSDFs. Wang et al. [21] proposed a method for pure specular light transport, which can also be used for glints rendering. Both of these approaches are significantly more technically involved than ours.

2.2 Path space regularization

Path space regularization renders scenes with difficult light paths by modifying the material parameters. Kaplanyan and Dachsbacher [22] introduced path space regularization for pure specular interactions. Bouchard et al. [23] also used regularization and introduced a custom MIS weight to select between unbiased and biased samplers. Jendersie and Grosch [24] extended the idea to microfacet models by manipulating the roughness parameter prior to the evaluation.

These methods cannot, in general, correct for the error introduced by regularization, other than by reducing the amount of regularization progressively during rendering. Compared to their work, our method can extrapolate from the regularized result to predict the result of original configurations, using gradient information.

2.3 Differentiable rendering methods

Differentiable rendering algorithms compute the derivative of a rendered image with respect to arbitrary scene parameters such as camera position, geometry, light sources, or material properties. This can be used for inverse rendering problems to optimize user-specified objective functions. Li et al. [25] proposed the first general-purpose differentiable path tracer, addressing visibility discontinuities with edge sampling. Loubet et al. [26] avoid edge sampling and improve performance with a reparameterization technique. Zhang et al. [27] introduced differentiable rendering for volume rendering, and Zhang et al. [28] proposed a path space

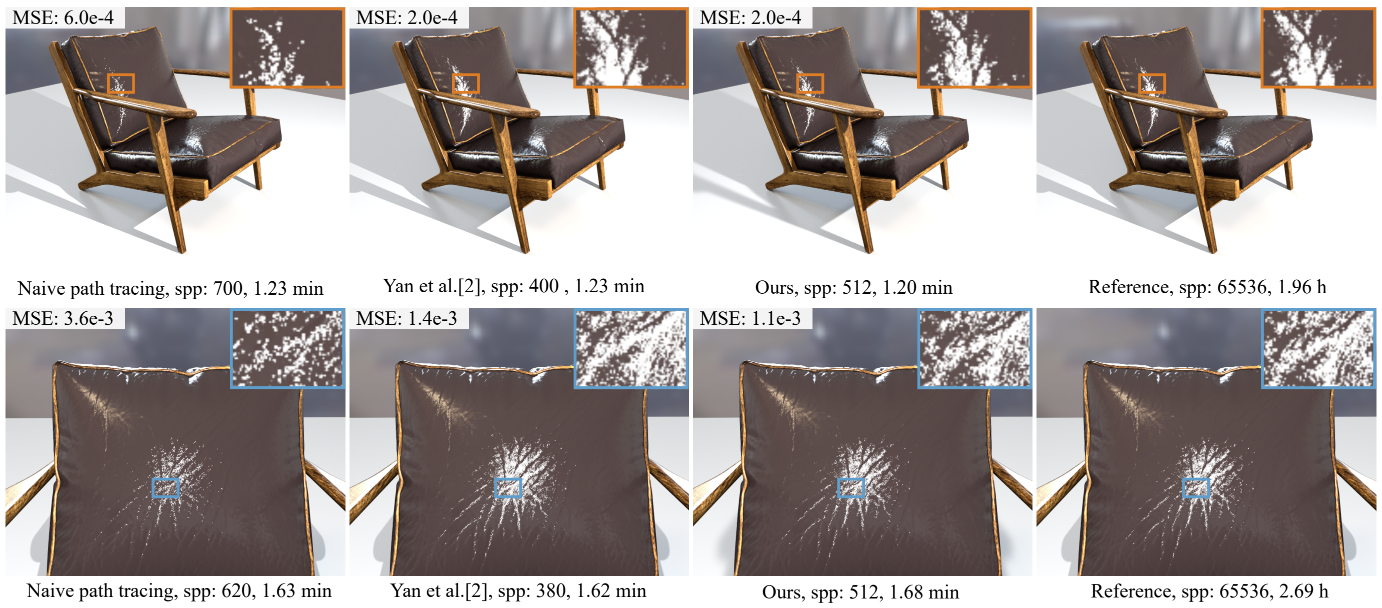

MSE: 6.0e-4
MSE: 2.0e-4
MSE: 2.0e-4
Naive path tracing, spp: 700, 1.23 min
Yan et al.[2], spp: 400 , 1.23 min
Ours, spp: 512, 1.20 min
Reference, spp: 65536, 1.96 h
MSE: 3.6e-3
MSE: 1.4e-3
MSE: 1.1e-3
Naive path tracing, spp: 620, 1.63 min
Yan et al.[2], spp: 380, 1.62 min
Ours, spp: 512, 1.68 min
Reference, spp: 65536, 2.69 h

Fig. 2. Comparison between our method (log-linear prediction model), naive path tracing and Yan et al. [2] with equal time and the reference. Our results are closer to the reference than both naive path tracing and Yan et al. [2] with equal time. Normal map: leather.

formulation of differentiable rendering. Nimier-David et al. [29] proposed a GPU-based differentiable rendering framework, and recently Nimier-David et al. [30] improved the scalability and efficiency of back-propagation within this framework. Please refer to Zhao et al. [31] for a summary and more details.

In our method, we use differentiable rendering for the rendered image with respect to BSDF roughness and light size. We use automatic differentiation tools from the Eigen library to compute the derivatives. The overhead is negligible, and other frameworks (like Mitsuba 2) could also be used. We do not need to consider the discontinuities caused by visibility, as geometry differentiability is not required in our method.

3 Our method

The key insight of our paper is to generate some paths which are easier to be sampled, by modifying the scene configuration, and then use the path contributions and gradients to predict the result for the actual scene configuration, as shown in Figure 1.

Our method has two steps. First, we perform a differentiable path tracing to render a scene with a larger light size and rougher surfaces (see Section 3.1), recording both the path contribution and the gradient with respect to light size and surface roughness (see Section 3.2); then we predict the actual contribution from the rendered result and the gradients (see Section 3.3).

Note that we focus on the direct illumination of the glints, same as previous works ([1] and [2]).

3.1 Differentiable BRDF and light source

As we discussed in Section 1, the difficulties of sampling in the glints rendering come from both the low-roughness specular surface and the sharp light source. To construct an easier sampling problem, a straightforward approach is making the surface rougher and the light source larger. Thus, we have the two factors for regularization: surface roughness α and light source size β. Both of these parameters (α and β) should be differentiable.

The surface roughness α is an input parameter for the BRDF using the microfacet model. A microfacet BRDF model is represented as:

$$\rho(\omega_i, \omega_o) = \frac{D_\alpha(h)F(\omega_o, h)G_\alpha(\omega_i, \omega_o)}{4(\omega_i \cdot n_g)(\omega_o \cdot n_g)} \tag{1}$$

where D_α represents the normal distribution function, $h = \frac{\omega_i+\omega_o}{|\omega_i+\omega_o|}$, G_α is the shadowing/masking term, F is the Fresnel term and n_g is the macro normal of the surface. Commonly used normal distribution functions include GGX and Beckmann distributions; the roughness α is differentiable in both of these functions. The G_α term is also affected by the roughness, and also differentiable. The Fresnel term is independent of roughness.

For the light source, we propose a Gaussian area light with a standard deviation parameter β to control the sharpness of the light. With smaller β, a sharp light effect can be achieved, while with a larger β, a smoother lighting (and easier sampling) is obtained. The Gaussian light is constructed on top of a rectangle, which is parameterized by (l_x, l_y) in local coordinates. The emitted radiance of position (l_x, l_y) is thus

$$E(l_x, l_y) = \frac{1}{2\pi\beta^2} e^{-(l_x^2+l_y^2)/(2\beta^2)}. \tag{2}$$

Other area light types (disk, sphere) could also be used, since gradients with respect to their radii can also be analytically derived. However, these derivations are more complex and out of our current scope.

3.2 Differentiable rendering

To get the gradient with respect to roughness and light size, we use automatic differentiation, using the autodiff component of the Eigen C++ library, though other similar tools could also be used.

Now that both the roughness and the light size are differentiable, we increase the roughness from α_t to α_s and the light size from β_t to β_s. Next, we perform path tracing with this modified configuration. We call the path sampled after modifying the scene configuration *source path*. After path tracing, we get the

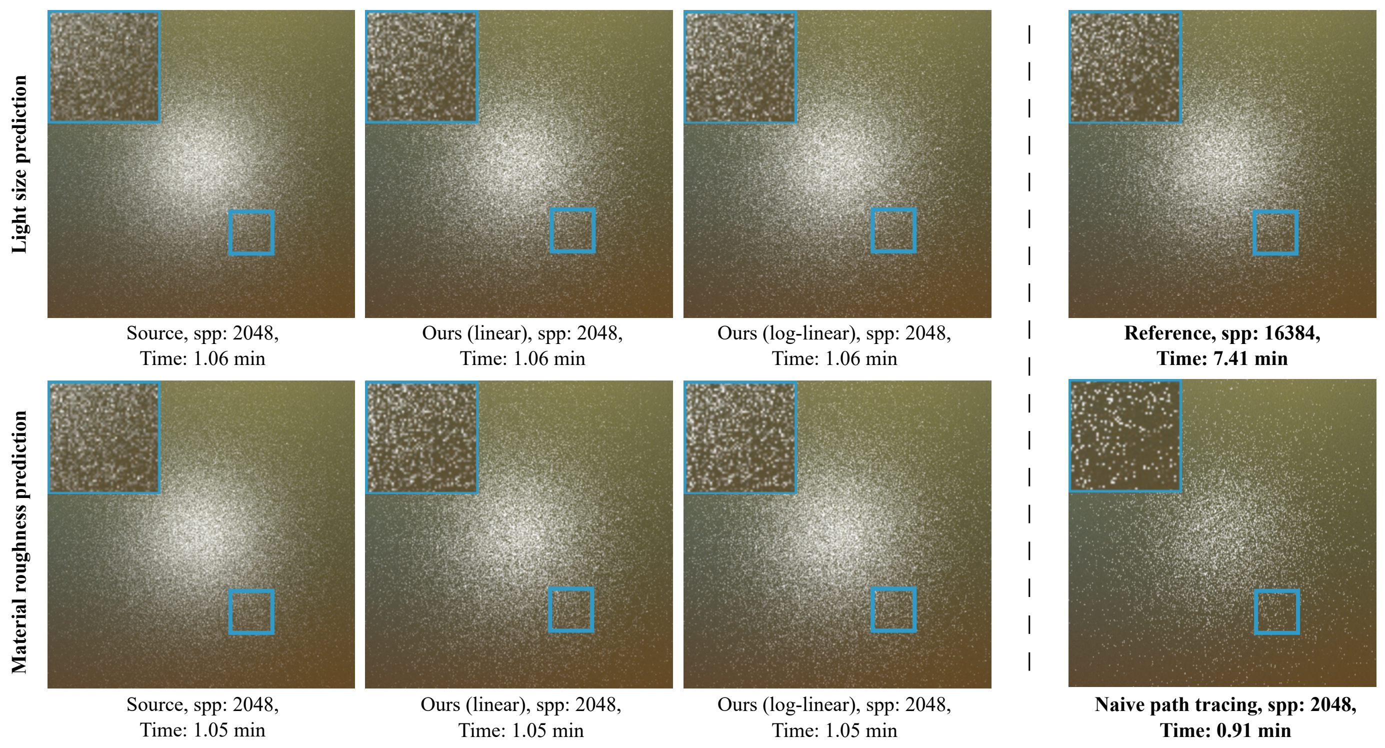

Light size prediction
Material roughness prediction
Source, spp: 2048, Time: 1.06 min
Ours (linear), spp: 2048, Time: 1.06 min
Ours (log-linear), spp: 2048, Time: 1.06 min
Reference, spp: 16384, Time: 7.41 min
Source, spp: 2048, Time: 1.05 min
Ours (linear), spp: 2048, Time: 1.05 min
Ours (log-linear), spp: 2048, Time: 1.05 min
Naive path tracing, spp: 2048, Time: 0.91 min

Fig. 3. Comparison between our method (linear prediction model), our method (log-linear prediction model), naive path tracing with equal spp and the reference. Compared with naive path tracing, both of our models can produce better results at an acceptable time cost. Normal map: isotropic noise.

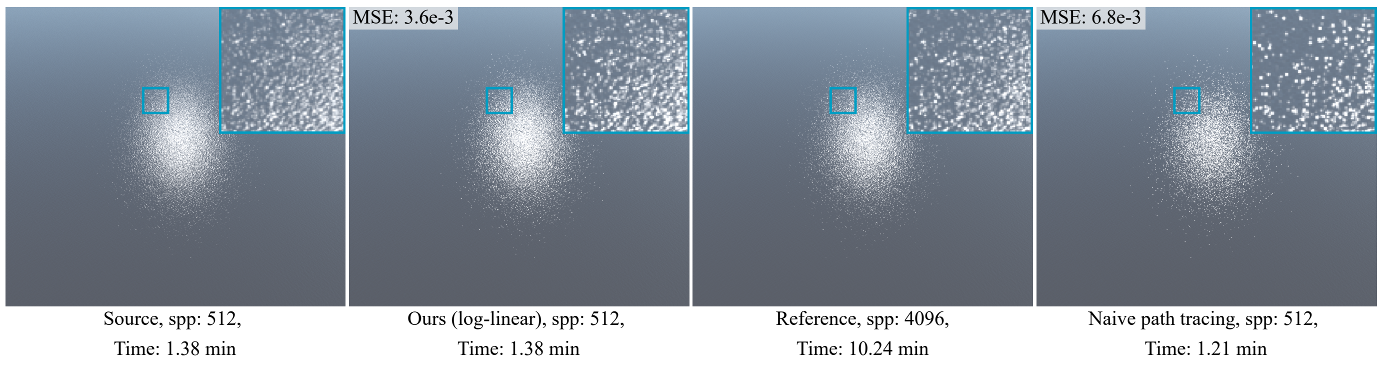

MSE: 3.6e-3
MSE: 6.8e-3
Source, spp: 512, Time: 1.38 min
Ours (log-linear), spp: 512, Time: 1.38 min
Reference, spp: 4096, Time: 10.24 min
Naive path tracing, spp: 512, Time: 1.21 min

Fig. 4. Comparison between rendered result with source configuration, our method (log-linear prediction model), naive path tracing with equal samples per pixel and the reference. Compared with naive path tracing, our method performs much better at an acceptable time cost. Normal map: wave.

contribution of each source path, accompanied with the gradient w.r.t. α and β, represented as J_α and J_β respectively.

BRDF importance sampling is an important component of efficient rendering. In our approach, we do not consider the derivative of pdf computation. Thus, we only compute the derivative of the BRDF value after sampling the outgoing direction, and divide it by the sampling pdf. As long as we divide the BRDF value by the correct pdf used for sampling, the value and gradient estimators remain unbiased. Note that multiple importance sampling is still performed, no different from standard path tracing. We do not need to consider discontinuities caused by object silhouettes, as differentiability with respect to geometry is not used in our method.

We accumulate the radiance and gradients from all source paths within a pixel and average them, obtaining an estimator of the pixel integral and its gradients. Note that the extrapolation is done per pixel (after accumulating value and gradients from all paths), not per path. Next, we use this pixel value and gradients to approximately reconstruct the pixel value under the original scene configuration in the next section.

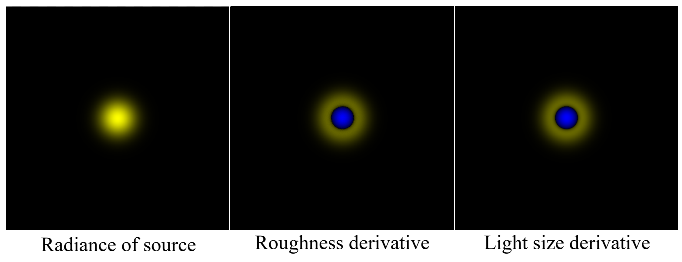


Fig. 5. The rendered image, derivative images w.r.t. the roughness and light size for a single glint. This scene includes a plane and a Gaussian light on the top. The positive and negative values are represented in green and blue color respectively.

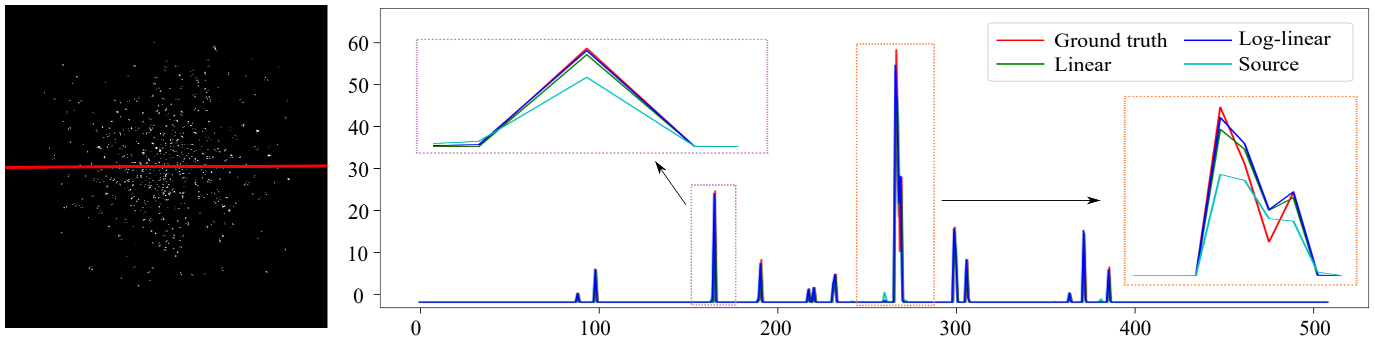


Fig. 6. Radiance curves as a function of a slice of pixel on the rendered image (left) between our method (linear prediction model), our method (log-linear prediction model), and the ground truth. For clarity, we use a scene with fewer bumps. The prediction is performed for light size, from 0.01 to 0.005.

In Figure 5, we show the rendered image, derivative image with respect to the roughness and derivative image w.r.t. the light size. We use large glints for clarity.

3.3 Glints regularization

Starting from the source pixel value R_s, and the gradients J_α and J_β with respect to surface roughness and light size, all computed at parameters (α_s,β_s), we predict the target pixel value R_t at parameters (α_t,β_t). The simplest approach is to assume that the pixel value function is locally linear, which yields the linear prediction:

$$R'_t = R_s + J_\beta(\beta_t - \beta_s), R_t = R'_t + J_\alpha(\alpha_t - \alpha_s) \qquad (3)$$

where β_t and β_s represents the light size of the target light and source light respectively. This is essentially using a first-order Taylor expansion to extrapolate. Note that the prediction can become negative, in which case it can simply be clamped to zero.

However, we found the linear prediction is not usually the best solution. We propose another prediction, log-linear, which is just applying a linear prediction (first-order Taylor expansion) to the logarithm of the pixel value. We do this in two steps, first applying a light size update, followed by a material roughness update:

$$R'_t = R_s * e^{J_\beta(\beta_t - \beta_s)/R_s}, R_t = R'_t * e^{J_\alpha(\alpha_t - \alpha_s)/R_s} \qquad (4)$$

The order of light size and material roughness updates in Equations 3 and 4 does not affect the result, thus we arbitrarily choose the light size first.

Both predictions can be used in practice, though in our experiments, we find the log-linear extrapolation usually produces a better fit to the ground truth. See Figure 6.

4 Results

We have implemented our algorithm inside the Mitsuba renderer [32]. All materials in our test scenes are using microfacet models [5] (using the Beckmann NDF): rough dielectric for the ocean scene, and rough conductor for others (coated with a smooth dielectric in the metallic paint example). We compare against Yan et al. [2] and naive path tracing with a large number of samples for accuracy validation. All timings in this section are measured on an Intel Xeon E5-2630@2.20GHz (20 cores) with 32 GB of main memory.

In this section, we first compare our method with a previous method and a path-traced reference for validation, and then analyze the effect of the main parameters in our model. Finally, we report the performance and error achieved by our method, compared to previous work.

In the following, by *source* we mean the regularized rendering with the increased light size and/or roughness, without extrapolation, while by *target* we mean the final extrapolated result with the desired light size and roughness. If the source roughness / light size is the same as the target, this means we do not perform extrapolation on this factor.

4.1 Quality validation

In this section, we compare our method against Yan et al. [2] and naive path tracing on several scenes. The reference is rendered by path tracing with a large number of samples per pixel. We use Mean Square Error (MSE) to measure the difference with the reference.

Bumps on flat plane. We first study the simple case of a plane with an isotropic noise normal map, rendered using environment lighting and a Gaussian light. In Figure 3, we compare our method with a directly rendered image with the same sampling rate and the ground truth rendered with more samples. By comparison, we find our method produces a less noisy result than the directly rendered image and our result is closer to the ground truth both visually and quantitatively. We also provide a video to show the temporal behavior and quality.

Leather on chair. The scene in Figure 2 shows a chair with two leather pillows, rendered using environment lighting and a Gaussian light. The leather pillows have a macro-level normal map and detailed microstructure bumps. The macro map covers 75cm × 75cm. The micro example normal map with resolution 512×512 covers 37.5mm×37.5mm. We compare our method (log-linear prediction) with naive path tracing and Yan et al. [2] with an equal sampling rate. Both our method and Yan et al. [2] produce results similar to the reference, while the naive path tracing misses a lot of glints.

Bent quad. Figure 7 shows a simple scene with a 5cm × 5cm bent quad with an isotropic noise normal map and a dielectric coating illuminated by a Gaussian light and an environment map. The resolution of the input isotropic noise normal map is 512×512, and covers about 0.71cm × 0.71cm. In this figure, we compare our result with naive path tracing with equal samples per pixel (spp), the reference and Yan et al. [2]. Our method produces result closer to the reference than both naive path tracing and Yan et al. [2], while the latter's memory cost is more than 1 GB larger than ours. Figure 8 shows the same scene configuration but with

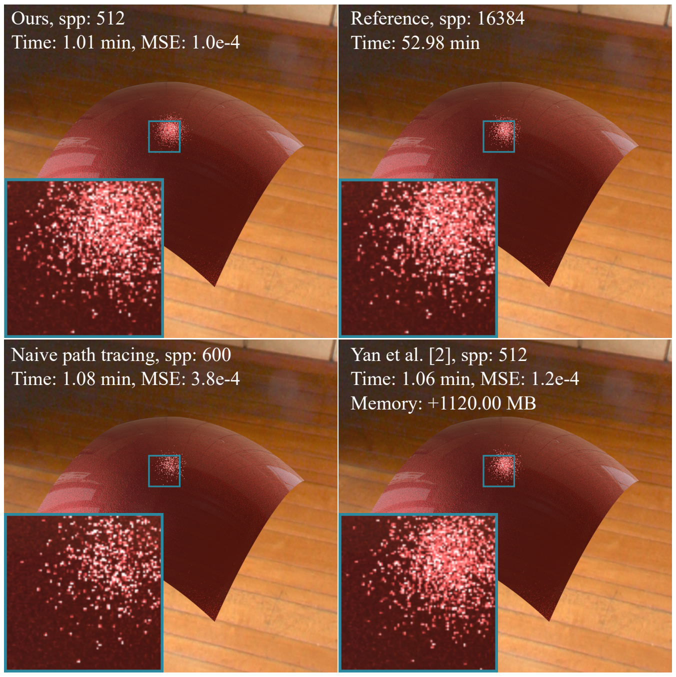

Ours, spp: 512
Time: 1.01 min, MSE: 1.0e-4
Reference, spp: 16384
Time: 52.98 min
Naive path tracing, spp: 600
Time: 1.08 min, MSE: 3.8e-4
Yan et al. [2], spp: 512
Time: 1.06 min, MSE: 1.2e-4
Memory: +1120.00 MB

Fig. 7. Comparison between our method (log-linear prediction model), naive path tracing with equal time, the reference and Yan et al. [2]. Our method produces results with higher quality than both naive path tracing and Yan et al. [2], and with lower memory cost. Normal map: isotropic noise with a coated layer.

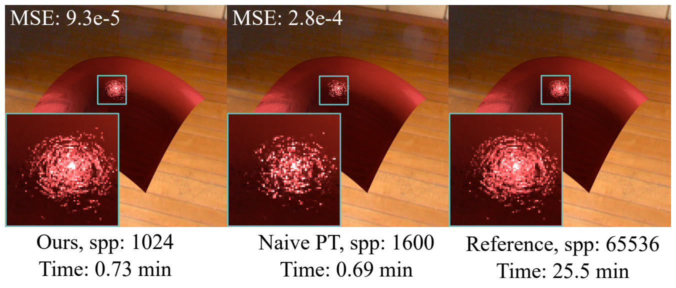


Fig. 8. Comparison between our method (log-linear prediction model), naive path tracing with equal time and the reference. Our method produces much better result than naive path tracing in equal time. Normal map: scratches with a coated layer.

a scratched normal map, showing that our method can handle not only bumpy surfaces, but also scratch effects, producing better results than naive path tracing with an equal spp.

Ocean waves. The scene in Figure 4 shows an ocean under a point light and environment lighting. We use a wave normal map with size $2K \times 2K$, with 10 tiles. We compare our method against naive path tracing with equal sampling rate. The result from naive path tracing does not find all the glints. We also show the result with the increased light size and roughness, which is too blurry compared to the reference.

Noise and bias. Our method introduces bias to the rendered results, at the expense of noise reduction. To better understand this trade-off, we compare the error between our method as a function of spp and the naive path tracing (equal spp), with respect to the reference (rendered with 65,536 spp) on the Chair scene in Figure 12. As shown in this figure, the error of our method decreases until about 8,192 spp and then keeps almost constant. The error comes from both variance and bias. When the spp reaches 8,192, the results are mostly converged, thus the error after 8,192 spp comes mostly from bias. In summary, with a low spp budget (for this scene, below 8,192 spp), our method is superior to the naive unbiased method both numerically and visually.

4.2 Parameter analysis

Prediction Model. In our method, we support two prediction models: linear and log-linear. We compare the results of these two models in Figure 3. It is not obvious which one works better, so we provide the radiance curve on a single scan-line of the image, as shown in Figure 6. Overall, the difference between the two methods is not very large, and both could be used in practice. We prefer the log-linear solution in our results, as its glint shapes appear more natural, and it never produces negative values that would need to be clamped.

Choice of source parameters. Glints rendering is challenging when the light source is sharp and the roughness is small. We use a larger light source and a rougher surface to predict the actual result. How should we choose the light size and the roughness? In Figure 9 and Figure 10, we analyze how to choose the source parameters.

- **Light size.** In Figure 9 we show the results with varying source light size. For the condition that target light size is 0.001, we find a source light size in nearby range is not helpful, while a slightly larger light size helps a little, like 0.005. When the light size reaches 0.01, the predicted results are the best. After further increasing the light size, the results are too blurry compared to the reference. For target light size 0.003, we may get a similar conclusion that source light size 0.006 helps a little, 0.015 and 0.03 looks the best and the larger ones become blurry.
- **Material roughness.** In Figure 10 we show the results of varying source material roughnesses. We find that it has the same law with light size prediction, but it's more sensitive to its parameter. When the target material roughness is 0.001, we find source value 0.005 works well. Then for target value 0.003, source value 0.006 looks the best.

In Figure 11, we show the MSE between our method and reference (rendered with 65,536 spp) as a function of light size and roughness on the Chair scene. The MSE is measured on cropped images (with resolution 64×64) from the Chair results. We find that the lowest error is achieved when source light size is set as 0.01 or source roughness set as 0.05. We find that increasing only light size or only roughness can be unpredictable; increasing both appears more robust.

Based on the results of all our experiments, we find that a source parameter of 5–10 times larger than the target for predicting light size, and 2–5 times larger for predicting roughness are the best choices. In our implementation, we set source light size with 0.01 for target light size 0.001, and source material roughness 0.005 for target material roughness 0.001.

4.3 Performance

In Table 1, we report all the scene settings, computation time and their error (MSE) with the reference for our test scenes. The cost of prediction is negligible compared to the rendering time (0.05 s for an image of $1K$ resolution), thus we ignored it in our report. We use log-linear model for all scenes in the table. Our method

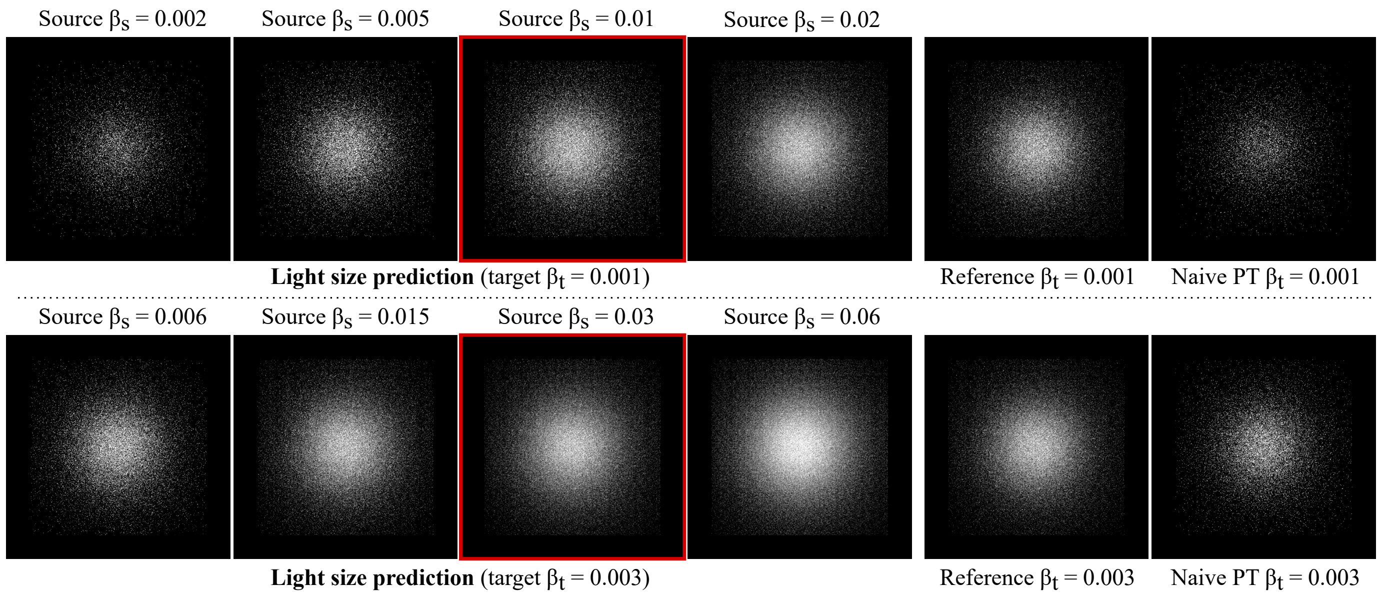

Source $\beta_s = 0.002$
Source $\beta_s = 0.005$
Source $\beta_s = 0.01$
Source $\beta_s = 0.02$
Light size prediction (target $\beta_t = 0.001$)
Reference $\beta_t = 0.001$
Naive PT $\beta_t = 0.001$
Source $\beta_s = 0.006$
Source $\beta_s = 0.015$
Source $\beta_s = 0.03$
Source $\beta_s = 0.06$
Light size prediction (target $\beta_t = 0.003$)
Reference $\beta_t = 0.003$
Naive PT $\beta_t = 0.003$

Fig. 9. Comparison between results predicted with different light size, naive path tracing with equal spp and the reference. *Source* means the result with increased light size, before extrapolation. The target light sizes are 0.001 (top) and 0.003 (bottom), and the target material roughness is 0.001. We conclude that increasing light size about 10 times is a good heuristic.

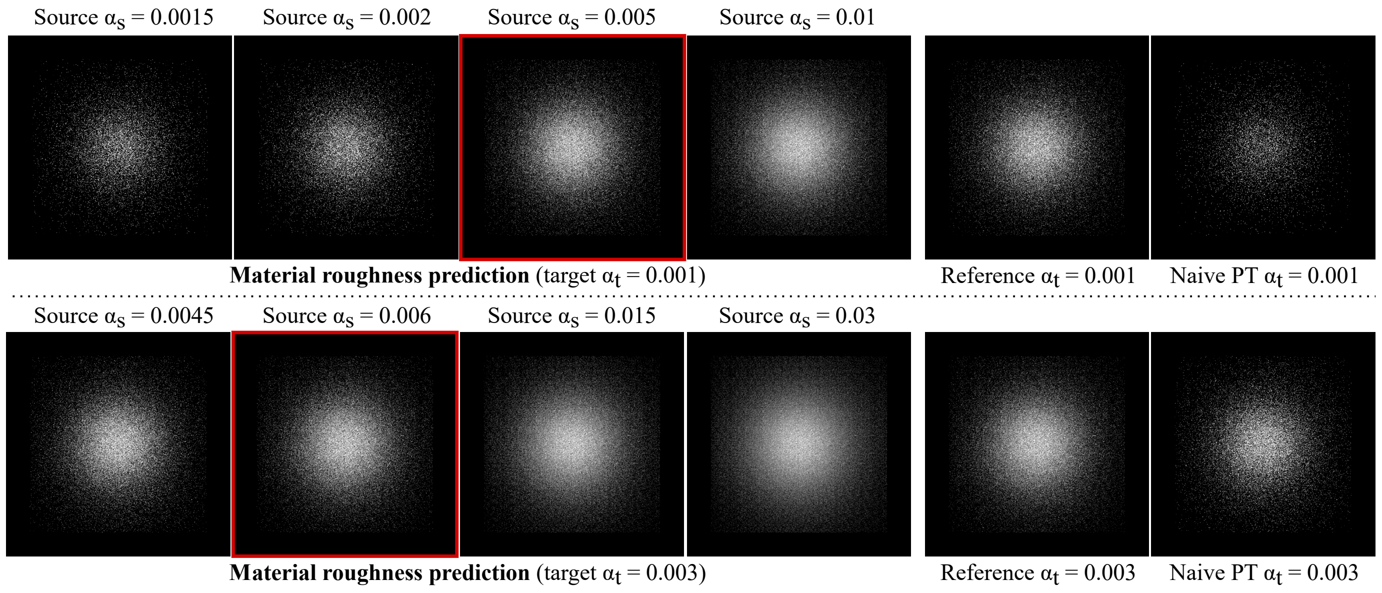

Source $\alpha_s = 0.0015$
Source $\alpha_s = 0.002$
Source $\alpha_s = 0.005$
Source $\alpha_s = 0.01$
Material roughness prediction (target $\alpha_t = 0.001$)
Reference $\alpha_t = 0.001$
Naive PT $\alpha_t = 0.001$
Source $\alpha_s = 0.0045$
Source $\alpha_s = 0.006$
Source $\alpha_s = 0.015$
Source $\alpha_s = 0.03$
Material roughness prediction (target $\alpha_t = 0.003$)
Reference $\alpha_t = 0.003$
Naive PT $\alpha_t = 0.003$

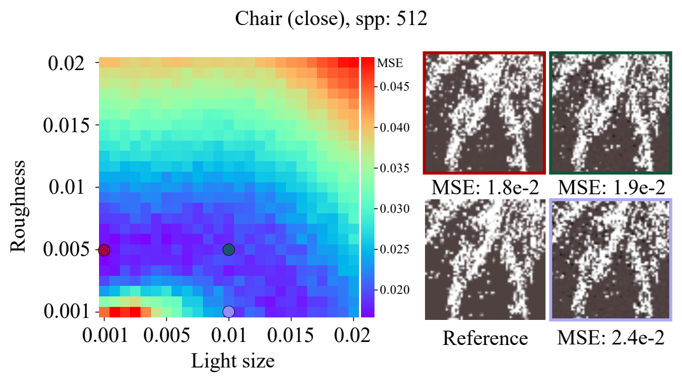
Fig. 10. Comparison among results predicted with different material roughness, naive path tracing with equal spp and the reference. *Source* means the result with increased roughness, before extrapolation. The target light size is 0.001 and the target material roughnesses are 0.001 (top) and 0.003 (bottom). We conclude that increasing roughness about 5 times is a good heuristic.

produces results with higher quality than naive path tracing with equal sampling rate (Bumpy Surface and Ocean scene) and equal time (Chair and Bent Quad) in most cases, which is confirmed by the MSE. Regarding the computational cost, our method takes a slightly longer time (about 13%) than naive path tracing with an equal sampling rate, due to the gradient tracking during path tracing. However, the result quality is improved significantly.

In Table 2, we provide the memory cost and MSE of our method and Yan et al. [2] with equal time. As shown in the table, Yan et al. [2] has higher computational memory cost than our method, due to its position-normal distribution and hierarchy, while our method does not rely on any of these extra structures. Regarding the error, our method produces same or higher quality in equal time in all cases. Furthermore, our method does not rely on ray differentials, making it straightforward to be extended to indirect glints, while Yan et al. [2] would require extra effort to accomplish multiple bounces.

4.4 Temporal coherence

Our method preserves temporal coherence. In the supplementary video, we compare our results with the extrapolation source (i.e., the rendering with larger roughness and light size that we extrapolate from) and the reference. Our method produces results with an equal amount of noise compared to the source. Therefore, if the source is already temporally coherent, our prediction will



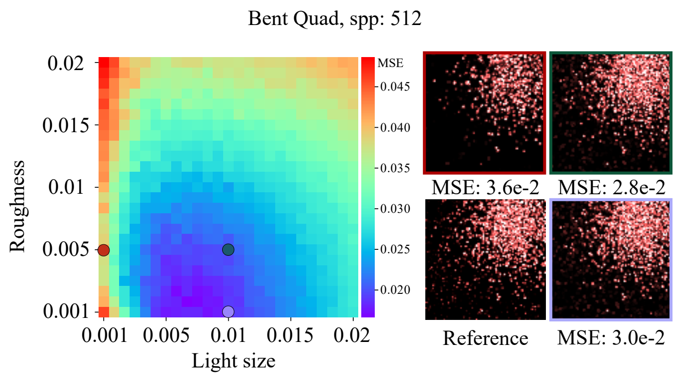


Fig. 11. MSE between our method and the reference as a function of light size and roughness, where the spp is set to 512 to compute the source rendered image. The errors are measured on cropped images (64×64) from the Chair scene (left) and the BentQuad scene (right). While it would have been ideal to increase only roughness in the left example, and increase only light size on the right, we have no way of knowing that up front, and these choices would not work well when reversed. This validates our heuristic of choosing 5× larger roughness and 10× larger light size, which appears to perform well in general.

TABLE 1
Scene settings, computation time and mean squared error (MSE) for our test scenes. Both β_t and α_t are set as 0.001 in all the test scenes. #Tri. is the count of triangles in the scene. Spp. represents sample per pixel for path tracing. All times are in minutes, and all results are predicted by the log-linear model. When β_t and β_s are the same, we do not perform the roughness predicting.

Scene	Fig.	Resolution	#Tri.	Normal map		Ours					Pt.			Reference	
				Size	Tile	α_s	β_s	Spp.	Time	MSE	Spp.	Time	MSE	Spp.	Time
Chair (far)	Fig. 2	1280 × 720	30.3K	512	20	0.001	0.01	512	1.20	**2.0e-4**	700	0.98	6.0e-4	4K	8.61
Chair (close)	Fig. 2	1280 × 720	30.3K	512	20	0.001	0.01	512	1.68	**1.1e-3**	620	1.47	3.6e-3	4K	12.71
Bumpy Surf. (top)	Fig. 3	512 × 512	3	512	13	0.001	0.01	2048	1.06	**1.3e-2**	512	0.91	2.7e-2	16K	7.41
Bumpy Surf. (bottom)	Fig. 3	512 × 512	3	512	13	0.01	0.001	2048	1.06	**1.3e-2**	512	0.91	2.4e-2	16K	7.41
Ocean	Fig. 4	1024 × 900	1	2048	10	0.001	0.01	512	1.38	**3.6e-3**	512	1.21	6.8e-3	4K	10.24
Bent Quad (iso. noise)	Fig. 7	1024 × 1024	19.6K	512	7	0.001	0.01	512	1.01	**1.0e-4**	600	0.88	3.8e-4	16K	52.98
Bent Quad (scratches)	Fig. 8	512 × 512	19.6K	512	7	0.001	0.01	512	0.25	**1.2e-4**	600	0.28	2.3e-4	32K	25.5

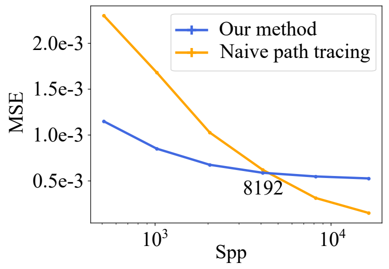


Fig. 12. MSE between our method or path tracing with the reference (rendered with 65,536 spp) over varying sampling rate on the Chair scene. The error of our method decreases until 8,192 spp and then almost keep constant after 8,192 spp. The error before 8,192 spp comes from both variance and bias. When the spp reaches 8,192, the results are converged, thus the error after 8,192 spp comes from bias.

TABLE 2
Memory cost and error comparison between our method and Yan et al. [2] with equal time. In this table, we show the memory overhead of Yan et al. [2] over our method. Note that the memory cost of Yan et al. [2] mainly depends on the resolution of the normal maps, the sampling rate in their Gaussian mixture construction and the storage cost of additional acceleration hierarchies.

Scene	Figure	Time	Ours		Yan et al. [2]		
			Spp.	MSE	Spp.	MSE	Memory cost
Chair (far)	Fig. 2	1.20	512	**2.0e-4**	400	2.0e-4	+1120.00 MB
Chair (close)	Fig. 2	1.68	512	**1.1e-3**	380	1.4e-3	+1120.00 MB
Bent Quad (iso.)	Fig. 7	1.01	512	**1.0e-4**	512	1.2e-4	+1120.00 MB

keep this property. However, if the source is still noisy and/or shimmering at a given sampling rate, our method cannot produce a better result by extrapolating from it. Still, our method can always produce more temporally coherent results than naive path tracing with the same spp.

4.5 Limitations and discussion

Our method produces convincing results for small glints, as shown in Figure 3. However, it produces more error for larger glints. In Figure 13, we show the predicted curve and the ground truth of one row in the image for large glints. We find that for small glints, our method is able to reconstruct the shape of ground truth very well (see Figure 6). However, large glints are more challenging. With the linear prediction model, the fall-off of the glints is not soft enough, while the log-linear model makes this fall-off too smooth, which yields overestimation for the off-peak area of the glints.

The main target configuration of our method is normal mapped materials with low roughness under sharp lighting, since naive sampling is difficult under this kind of configuration. When the roughness or light source increases, path tracing is able to produce noise-free results with reasonable sampling rate, and as expected, our method loses its benefits, as shown in Figure 14.

5 Conclusion

We have presented a method using differentiable rendering, and applied it to glints rendering. As far as we know, this is the first method to use differentiable rendering for regularization. Our method is simple to implement, and does not require any invasive

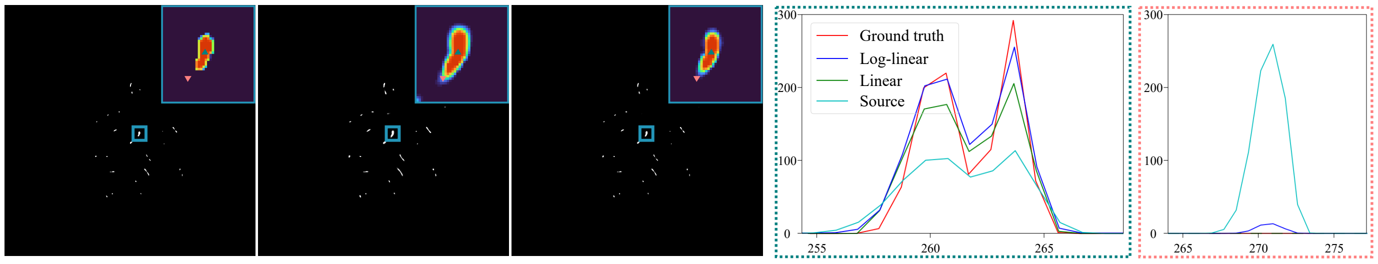


Fig. 13. (a) Comparison of our linear and log-linear model. The predicted pixel values of the glint are shown with color map for better visualization. (b) The radiance curves for a slice of pixels through the green triangle, which is the center a glint. The log-linear curve appears closer to the ground truth curve. (c) On a different scanline not passing through the glint center, the log-linear solution overestimates the ground truth. Overall, the difference between the two methods is not large, and both can be used in practice. We prefer the log-linear solution, as its glint shapes appear more natural, and it never produces negative values that would need to be clamped.

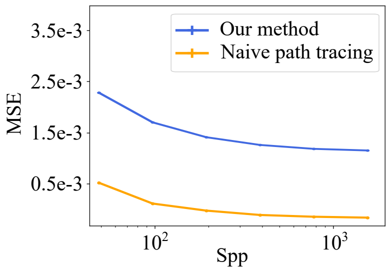


Fig. 14. MSE between our method or path tracing with the reference (rendered with 4,096 spp) over varying sampling rate on the Chair scene, where both α_t and β_t are set as 0.01. When the roughness or light source are large, path tracing is able to produce noise-free results with reasonable sampling rate, and our method loses its benefit.

changes to the rendering method, except to compute the derivatives of BRDF roughness and light size. Our method shows less noise than the naive path tracing on several microstructures, and is able to preserve the temporal coherence, though it does introduce bias. Our method is an easy fit into the multiple importance sampling framework and can be potentially used in any Monte Carlo path sampling algorithms, although we only demonstrate on standard path tracing. We show our method with Gaussian light, however, it could support any differentiable light sources.

In the future, we are interested in improving it further, by using a better prediction model to have a better fit to the ground truth, potentially utilizing deep learning or other higher-order models, like a log-quadratic model. Using differentiable regularization for other applications, such as caustics, is also an interesting research direction. Extending direction illumination to indirect illumination will also be an interesting extension, which could be done by tracing the gradients for each material roughness along the path.

Acknowledgments

We thank the editors and reviewers for the valuable comments and suggestions. This work has been partially supported by the National Natural Science Foundation of China under grant No. 62172220 and 61802187, the Fundamental Research Funds for the Central Universities under grant No. 30920021133 and the China Postdoctoral Science Foundation under grant No. 2020M671500. Ling-Qi Yan is supported by gift funds from Adobe, Dimension 5 and XVerse.

References

[1] L.-Q. Yan, M. Hašan, W. Jakob, J. Lawrence, S. Marschner, and R. Ramamoorthi, "Rendering glints on high-resolution normal-mapped specular surfaces," *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2014)*, vol. 33, no. 4, 2014.

[2] L.-Q. Yan, M. Hašan, S. Marschner, and R. Ramamoorthi, "Position-normal distributions for efficient rendering of specular microstructure," *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2016)*, vol. 35, no. 4, 2016.

[3] K. E. Torrance and E. M. Sparrow, "Theory for off-specular reflection from roughened surfaces," *Josa*, vol. 57, no. 9, pp. 1105–1114, 1967.

[4] P. Beckmann and A. Spizzichino, "The scattering of electromagnetic waves from rough surfaces," *Norwood, MA, Artech House, Inc., 1987, 511 p.*, 1987.

[5] B. Walter, S. R. Marschner, H. Li, and K. E. Torrance, "Microfacet models for refraction through rough surfaces," in *Proceedings of the 18th Eurographics conference on Rendering Techniques*, 2007, pp. 195–206.

[6] L.-Q. Yan, M. Hašan, B. Walter, S. Marschner, and R. Ramamoorthi, "Rendering specular microgeometry with wave optics," *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2018)*, vol. 37, no. 4, 2018.

[7] B. Wang, M. Hašan, N. Holzschuch, and L.-Q. Yan, "Example-based microstructure rendering with constant storage," *ACM Transactions on Graphics*, vol. 39, no. 5, pp. 1–12, 2020.

[8] A. Kuznetsov, M. Hašan, Z. Xu, L.-Q. Yan, B. Walter, N. K. Kalantari, S. Marschner, and R. Ramamoorthi, "Learning generative models for rendering specular microgeometry," *ACM Trans. Graph.*, vol. 38, no. 6, 2019.

[9] J. Zhu, Y. Xu, and L. Wang, "A stationary svbrdf material modeling method based on discrete microsurface," *Computer Graphics Forum (Proceedings of Pacific Graphics 2019)*, vol. 38, no. 7, pp. 745–754.

[10] W. Jakob, M. Hašan, L.-Q. Yan, J. Lawrence, R. Ramamoorthi, and S. Marschner, "Discrete stochastic microfacet models," *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2014)*, vol. 33, no. 4, 2014.

[11] B. Wang, L. Wang, and N. Holzschuch, "Fast global illumination with discrete stochastic microfacets using a filterable model," *Computer Graphics Forum (Proceedings of Pacific Graphics 2018)*, vol. 37, no. 7, 2018.

[12] B. Wang, H. Deng, and N. Holzschuch, "Real-time glints rendering with pre-filtered discrete stochastic microfacets," *Computer Graphics Forum*, vol. 39, no. 6, pp. 144–154, 2020.

[13] X. Chermain, B. Sauvage, J.-M. Dischler, and C. Dachsbacher, "Procedural physically based brdf for real-time rendering of glints," *Computer Graphics Forum*, vol. 39, no. 7, pp. 243–253, 2020.

[14] X. Chermain, S. Lucas, B. Sauvage, J.-M. Dischler, and C. Dachsbacher, "Real-time geometric glint anti-aliasing with normal map filtering," *Proc. ACM Comput. Graph. Interact. Tech.*, vol. 4, no. 1, Apr. 2021.

[15] B. Raymond, G. Guennebaud, and P. Barla, "Multi-scale rendering of scratched materials using a structured sv-brdf model," *ACM Transactions on Graphics*, vol. 35, no. 4, pp. 57:1–57:11, 2016.

[16] S. Werner, Z. Velinov, W. Jakob, and M. B. Hullin, "Scratch iridescence: Wave-optical rendering of diffractive surface structure," *ACM Transac-*

tions on Graphics (Proceedings of SIGGRAPH Asia 2017), vol. 36, no. 6, pp. 207:1–207:14, 2017.

[17] Z. Velinov, S. Werner, and M. B. Hullin, "Real-time rendering of wave-optical effects on scratched surfaces," *Computer Graphics Forum (Proc. of EUROGRAPHICS 2018)*, vol. 37, no. 2, 2018.

[18] T. Zirr and A. S. Kaplanyan, "Real-time rendering of procedural multi-scale materials," in *Proceedings of the 20th ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*. ACM, 2016, pp. 139–148.

[19] T. Zeltner, I. Georgiev, and W. Jakob, "Specular manifold sampling for rendering high-frequency caustics and glints," *Transactions on Graphics (Proceedings of SIGGRAPH)*, vol. 39, no. 4, Jul. 2020.

[20] X. Chermain, B. Sauvage, J.-M. Dischler, and C. Dachsbacher, "Importance Sampling of Glittering BSDFs based on Finite Mixture Distributions," in *Eurographics Symposium on Rendering - DL-only Track*, A. Bousseau and M. McGuire, Eds. The Eurographics Association, 2021.

[21] B. Wang, M. Hašan, and L.-Q. Yan, "Path cuts: Efficient rendering of pure specular light transport," *Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 2020.

[22] A. S. Kaplanyan and C. Dachsbacher, "Path Space Regularization for Holistic and Robust Light Transport," *Computer Graphics Forum*, 2013.

[23] G. Bouchard, J.-C. Iehl, V. Ostromoukhov, and P. Poulin, "Improving robustness of monte-carlo global illumination with directional regularization," in *SIGGRAPH Asia 2013 Technical Briefs*. Association for Computing Machinery, 2013.

[24] J. Jendersie and T. Grosch, "Microfacet Model Regularization for Robust Light Transport," *Computer Graphics Forum*, 2019.

[25] T.-M. Li, M. Aittala, F. Durand, and J. Lehtinen, "Differentiable monte carlo ray tracing through edge sampling," *ACM Trans. Graph.*, vol. 37, no. 6, Dec. 2018.

[26] G. Loubet, N. Holzschuch, and W. Jakob, "Reparameterizing discontinuous integrands for differentiable rendering," *Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, vol. 38, no. 6, Dec. 2019.

[27] C. Zhang, L. Wu, C. Zheng, I. Gkioulekas, R. Ramamoorthi, and S. Zhao, "A differential theory of radiative transfer," *ACM Trans. Graph.*, vol. 38, no. 6, pp. 227:1–227:16, 2019.

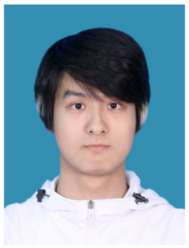
[28] C. Zhang, B. Miller, K. Yan, I. Gkioulekas, and S. Zhao, "Path-space differentiable rendering," *ACM Trans. Graph.*, vol. 39, no. 4, pp. 143:1–143:19, 2020.

[29] M. Nimier-David, D. Vicini, T. Zeltner, and W. Jakob, "Mitsuba 2: A retargetable forward and inverse renderer," *ACM Trans. Graph.*, vol. 38, no. 6, Nov. 2019.

[30] M. Nimier-David, S. Speierer, B. Ruiz, and W. Jakob, "Radiative backpropagation: An adjoint method for lightning-fast differentiable rendering," *Transactions on Graphics (Proceedings of SIGGRAPH)*, vol. 39, no. 4, Jul. 2020.

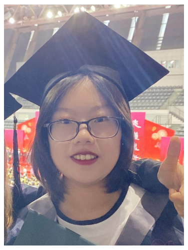
[31] S. Zhao, W. Jakob, and T.-M. Li, "Physics-based differentiable rendering: From theory to implementation," in *ACM SIGGRAPH 2020 Courses*, ser. SIGGRAPH 2020. Association for Computing Machinery, 2020.

[32] W. Jakob, "Mitsuba renderer," http://www.mitsuba-renderer.org/, 2010.



Jiahui Fan is a PhD candidate at School of Computer Science and Engineering in NJUST, working on deep learning and its application in photo-realistic rendering.

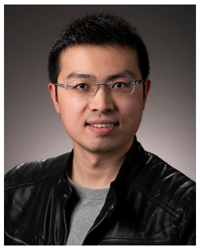
Beibei Wang received the PhD degree from Shandong University, in 2014 and visited Telecom ParisTech from 2012 to 2014. She is an associate professor with the Nanjing University of Science and Technology. She worked as a postdoc with Inria from 2015 to 2017. She joined NJUST in March 2017. Her research interests include rendering and game development.



Wenshi Wu is a master student at Nanjing University of Science and Technology. She received her bachelor degree from the same university in 2021. Her research interest mainly includes participating media rendering.

Miloš Hašan is a Senior Research Scientist at Adobe Research in San Jose. His research focuses on several areas of computer graphics and rendering, including light transport simulation, material reflectance models, and inverse rendering problems including material capture; he also contributes to production rendering systems. He received his Ph.D. in Computer Science from Cornell University in 2009.

Jian Yang received the Ph.D. degree in pattern recognition and intelligence systems from the Nanjing University of Science and Technology (NJUST) in 2002. In 2003, he was a Post-Doctoral Researcher with the University of Zaragoza. From 2004 to 2006, he was a Post-Doctoral Fellow with the Biometrics Centre, The Hong Kong Polytechnic University. From 2006 to 2007, he was a Post-Doctoral Fellow with the Department of Computer Science, New Jersey Institute of Technology. He is currently a Changjiang Distinguished Professor with the School of Computer Science and Technology, NJUST. His research interests include pattern recognition, computer vision, and machine learning. He is also an Associate Editor of Pattern Recognition Letters and the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS.



Ling-Qi Yan is an assistant professor of Computer Science at UC Santa Barbara, co-director of the MIRAGE Lab, and affiliated faculty in the Four Eyes Lab. Before that, he received his doctor degree from the Department of Electrical Engineering and Computer Sciences at UC Berkeley and obtained his bachelor degree in Computer Science from Tsinghua University. His research interests include physically-based rendering, real-time ray tracing and realistic appearance modeling.