Noise analysis of a synthetically rendered scene in sensor-realistic image simulation

Christian Kludt[1], Frederik Seiler[2], Verena Eichinger[2], Johannes Meyer[1], Ira Effenberger[2], Thomas Längle[1], and Jürgen Beyerer[1]

[1] Fraunhofer IOSB, Fraunhoferstraße 1, 76131 Karlsruhe
[2] Fraunhofer IPA, Nobelstraße 12, 70569 Stuttgart

Abstract This paper investigates how the noise characteristics of synthetically generated camera images correspond to those of a real camera. We determine the photon transfer curve from a set of rendered images of a static scene. Furthermore, we present a method to identify the regions with high temporal noise, i.e., rendering noise, in synthetically generated data from a single rendered image. Finally, we present a strategy on how a parameterization of the rendering can be achieved that minimizes the noise while also minimizing the rendering time.

Keywords Synthetic data generation, sensor-realistic simulation, noise analysis, EMVA 1288

1 Introduction

The advances in detecting and classifying defects that we might expect from machine learning (ML) approaches have often been stymied by lack of data. To train the AI models, they would need to be fed with a large number of examples of good products, but also supplied with precisely labeled bad ones. There are simply not enough of those, if any at all, which is why we turned the focus of our efforts on synthetic image generation as it has been performed in various visual inspection applications [1–8]. The idea is to simulate the entire testing and inspection environment – specimen geometry, material properties, lighting, sensor technology – to produce images that are synthetic, but still sufficiently realistic. And then we can use data of defects that we have

DOI: 10.58895/ksp/1000174496-8 erschienen in:
Forum Bildverarbeitung 2024
DOI: 10.58895/ksp/1000174496 | https://www.ksp.kit.edu/site/books/m/10.58895/ksp/1000174496/

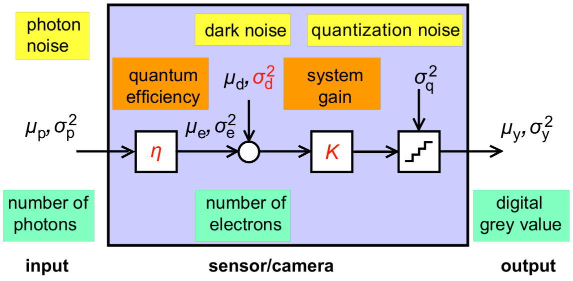


Figure 1: Mathematical camera model of a single pixel (source: EMVA Standard 1288 [9]).

gathered in the past to add synthetic defects as well and vary them in various ways. This might help us solve the "chicken or egg" problem. ML-based reproduction of images with defects requires that there are at least some images available, so it still depends on the quantity and quality of the input data. We can also build in any kind of defect we want, and, of course, the synthetic images created in this way are always labeled perfectly.

In the following, we analyze the noise characteristics of such a synthetic scene and how it resembles the linear camera model according to the standard EMVA 1288 [9]. Furthermore, we are developing strategies on how the noise characteristics can be improved so that they more closely resemble those of a real camera.

2 Fundamentals

2.1 Image Formation

We assume the transmission system to be a linear, shift invariant system. A standard digital industrial camera provides a linear photo response characteristic: the digital signal increases linearly with the number of photons received. These assumptions describe the properties of an ideal camera or sensor as described by the EMVA Standard 1288 (cf. Fig. 1) [9,10].

When a mean number of photons μ_p reaches the pixel area during exposure time, the fraction η is absorbed and creates a mean number of photo electrons

$$\mu_\mathrm{e} = \eta \mu_\mathrm{p}. \tag{1}$$

The dark current μ_d is the mean number of electrons present without light. It is added to the mean number of electrons μ_e. Together they form a charge, which is converted by a capacitor to a voltage and amplified by the system gain K. Then the voltage is digitized resulting in a digital gray value μ_y:

$$\mu_\mathrm{y} = K(\mu_\mathrm{e} + \mu_\mathrm{d}) = K\mu_\mathrm{e} + \mu_\mathrm{y.dark}. \tag{2}$$

The mean photon flux fluctuates randomly according to the Poisson probability distribution [11]. Therefore, the variance of the electron noise is equal to the mean number of electrons:

$$\sigma_\mathrm{e}^2 = \mu_\mathrm{e}. \tag{3}$$

All noise sources related to the sensor read out and amplifier circuits can be described by a signal independent normally distributed noise source with variance σ_d^2. The final analog-to-digital conversion adds another noise source that is uniformly distributed with variance $\sigma_\mathrm{q}^2 = 1/12$. Because the variances of all noise sources add up linearly, the total temporal variance of the digital signal μ_y is given according to the laws of error propagation by

$$\sigma_\mathrm{y}^2 = K^2(\sigma_\mathrm{d}^2 + \sigma_\mathrm{e}^2) + \sigma_\mathrm{y}. \tag{4}$$

After plugging (3) into (4), we get

$$\sigma_\mathrm{y}^2 = K^2(\sigma_\mathrm{d}^2 + \mu_\mathrm{e}) + \sigma_\mathrm{y}. \tag{5}$$

The mean number of photo electrons μ_e cannot be measured. From (2) we get

$$\mu_\mathrm{e} = (\mu_\mathrm{y} - \mu_\mathrm{y.dark})/K. \tag{6}$$

Now plugging (6) into (5) yields

$$\sigma_\mathrm{y}^2 = K^2\sigma_\mathrm{d}^2\sigma_\mathrm{q}^2 + K(\mu_\mathrm{y} - \mu_\mathrm{y.dark}). \tag{7}$$

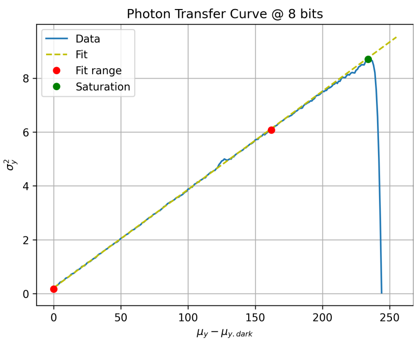


Figure 2: Photon transfer function of a real camera. The graph draws the measured variance σ_{y}^2 versus the mean photo-induced gray values $\mu_{\mathrm{y}} - \mu_{\mathrm{y.dark}}$ and the linear regression line used to determine the overall system gain K. The red dots mark the 0—70% range of saturation that is used for the linear regression.

Now the unknown parameters from Fig. 1 (red color) can be determined using the so called photon transfer method [12]: The system gain K is determined from the slope of (7), and the dark noise variance σ_{d}^2 from its offset.

So in summary, for a linear camera, the temporal noise with variance σ_{y}^2 shows a linear dependence on the mean signal μ_{y}. In order to verify whether a camera or a (synthetically generated) dataset exhibits this linear characteristic, one simply has to apply the photon transfer method and analyze the linearity of the graph. A real camera has the characteristics as shown in Fig. 2.

2.2 Ray Tracing

Ray tracing is a rendering technique that simulates the behavior of light to create realistic images using geometric optics. At its core, the process begins by sending rays from a virtual camera into a scene. When a ray encounters an object, the algorithm evaluates how light interacts with the surface at that point. This involves calculating surface normals, material properties, and the angle of incidence, which inform how much light is reflected, refracted, or absorbed by the material. Finally, after processing all rays for each pixel, the results are combined to form the final image. The accumulated energy values, influenced by lighting and surface properties, create a 2D image representation of the scene. During the rendering, the render equation is approximately solved using a monte carlo approach. The render equation

$$L_{\mathrm{o}}(\mathbf{p}, \boldsymbol{\omega}_{\mathrm{o}}) = L_{\mathrm{e}}(\mathbf{p}, \boldsymbol{\omega}_{\mathrm{o}}) + \int_{\Omega} \mathrm{BRDF}(\mathbf{p}, \boldsymbol{\omega}_{\mathrm{i}}, \boldsymbol{\omega}_{\mathrm{o}}) L_{\mathrm{i}}(\mathbf{p}, \boldsymbol{\omega}_{\mathrm{i}}) (\boldsymbol{\omega}_{\mathrm{i}}^{\mathsf{T}} \mathbf{n}) \, \mathrm{d}\, \boldsymbol{\omega}_{\mathrm{i}} \tag{8}$$

describes the propagation of light through the scene with $L_{\mathrm{o}}(\mathbf{p}, \boldsymbol{\omega}_{\mathrm{o}})$ representing the outgoing radiance from point $\mathbf{p}$ in the direction $\boldsymbol{\omega}_{\mathrm{o}}$, $L_{\mathrm{e}}(\mathbf{p}, \boldsymbol{\omega}_{\mathrm{o}})$ being the emitted radiance from point $\mathbf{p}$ in the direction $\boldsymbol{\omega}_{\mathrm{o}}$, $\mathrm{BRDF}(\mathbf{p}, \boldsymbol{\omega}_{\mathrm{i}}, \boldsymbol{\omega}_{\mathrm{o}})$ denoting the bidirectional reflectance distribution function (BRDF), which indicates how light is reflected from the direction $\boldsymbol{\omega}_{\mathrm{i}}$ to the direction $\boldsymbol{\omega}_{\mathrm{o}}$ at point $\mathbf{p}$, $L_{\mathrm{i}}(\mathbf{p}, \boldsymbol{\omega}_{\mathrm{i}})$ representing the incoming radiance to point $\mathbf{p}$ from the direction $\boldsymbol{\omega}_{\mathrm{i}}$, the cosine $\boldsymbol{\omega}_{\mathrm{i}}^{\mathsf{T}} \mathbf{n}$ of the angle of incidence between the incoming light direction $\boldsymbol{\omega}_{\mathrm{i}}$ and the surface normal $\mathbf{n}$ and the positive hemisphere Ω above point $\mathbf{p}$.

3 Setup

The pipeline for image synthesis consists of several steps, which are described subsequently. First, the setup for the real world image acquisition is virtually recreated using 3D models. The open source 3D software Blender [13] makes it possible to either model the required objects manually or import existing models from CAD data and other sources. In Blender the visual inspection system setup is recreated in detail and the positions of sensors and lighting can be defined in the 3D scene. To generate images from the 3D scene, ray tracing is used as a rendering method. Ray tracing physically simulates light rays to create photo-realistic images by following the path of light rays and analyzing their interaction with surfaces to calculate effects such as shadows and reflections. Mitsuba 3 [14] is used as the rendering engine. To render the scene created in Blender with Mitsuba, all objects in the scene are exported and the Mitsuba Blender Add-On is used to generate an XML scene description. This scene description contains a textual description of the recreation of the measuring setup including all the information required for rendering with Mitsuba 3. The rendering is followed by a denoising process using Intel Open Image Denoise [15]. This open source library offers high-quality and high-performance denoising filters. The rendering and denoising process is automated through scripting. In the process of rendering an image using the Mitsuba rendering tool, each generated pixel value represents the energy received at that pixel. This energy is linearly assigned to the pixel values. Similar to the saturation in CCD sensors, the pixel values are clipped at a maximum value of one. Before saving the images, a gamma correction of 2.2 is applied in accordance with the sRGB color space, and the images are quantized into 8-bit formats.

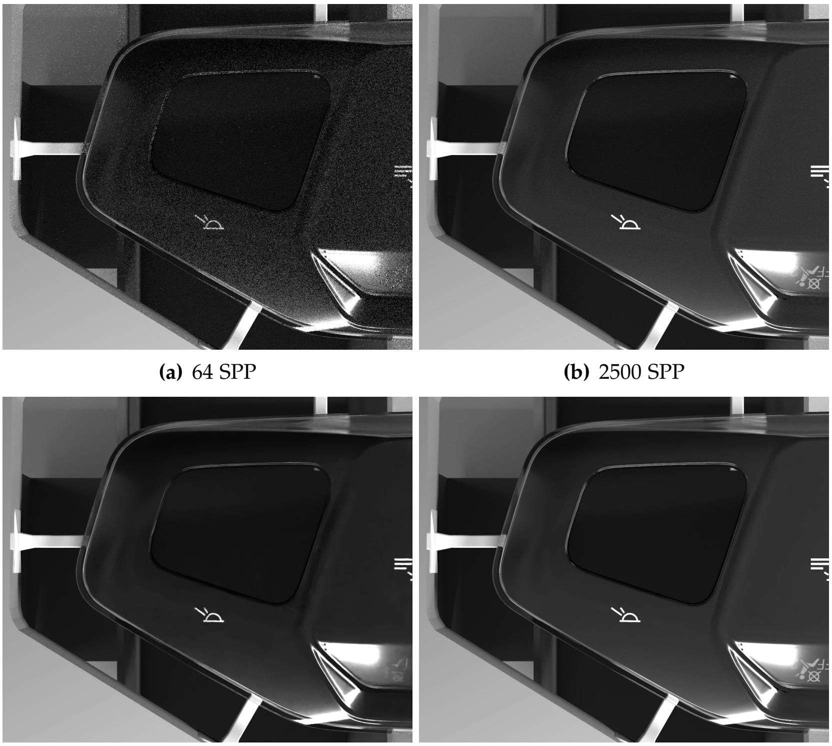
The image generation in ray tracing algorithms is dependent on random variables. Therefore, the seed of the random number generator is changed to produce statistically independent images.

The stochastic nature of the rendering process leads to local deviations from the perfect scene, which can be interpreted as spatial noise. Less noise can be achieved by increasing the rendering parameter samples per pixel (SPP) but at the cost of higher rendering times. In practice, the maximum allowed time to render an image sets an upper boundary for the maximum SPP.

Especially in dark field setups the noise is very strong. Fig. 3 depicts the test object as seen by the virtual camera, with the effects of different SPP and the denoiser turned off or on.

4 Experiments

The scene is a dark field setup and consists of several components, with the base being a housing made of aluminum profiles and black cover plates that ensure controlled imaging conditions on the inside. An area light is installed at the bottom of the housing, above which a movable shutter is fitted to shade the light. Additional lights are positioned above the mount on the rear wall and on both sides, whereby these area lights illuminate the test object from three directions. The camera is positioned above the test object. The object under consideration is a two-component injection molded part for which CAD data is provided. The top layer consists of transparent polymethyl methacrylate (PMMA), under which symbols with varying degrees of transparency are arranged. The next layer represents a deformed film, while the base consists of a thermoplastic base body. Fig. 4 visualizes



(a) 64 SPP **(b)** 2500 SPP

(c) 64 SPP, denoised **(d)** 2500 SPP, denoised

Figure 3: Scene rendered with different samples per pixel (SPP) and denoiser turned off or on.

the composition of the inspected object. For the simulation of image data, suitable materials for all components are defined using surface scattering models. The Mitsuba 3 renderer provides a principled bidirectional scattering distribution function (BSDF) model that can cover a wide range of materials and is used to simulate all materials contained in the scene. The individual parameters are adjusted, as far as possible, according to the real material properties, such as the refractive index of PMMA. Where it is not possible to transfer the material properties directly to the simulation model, the parameters are selected in such a

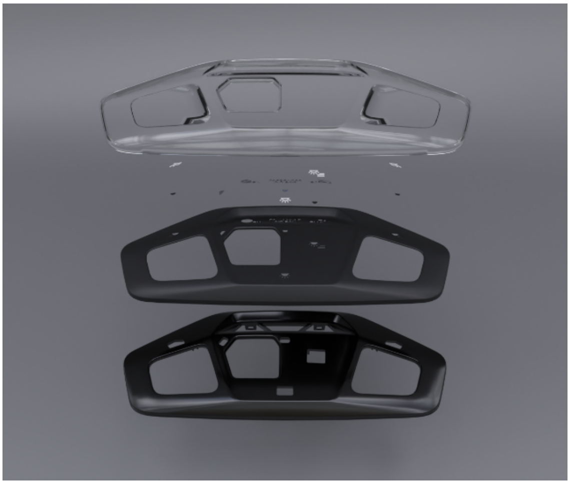
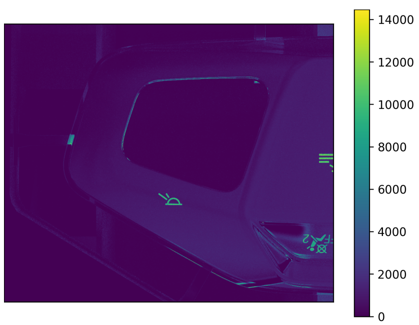


Figure 4: Composition of the modeled product.

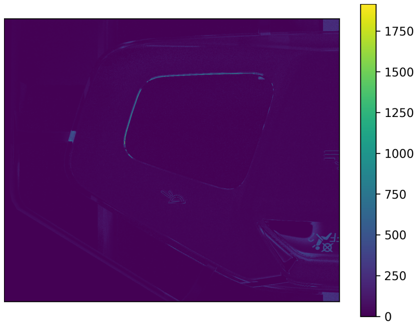
way that the visual impression of the rendered images closely matches the real appearance.

For the evaluations, 50 images each with 64 SPP and 2500 SPP are generated using the pipeline described in Sec. 3.

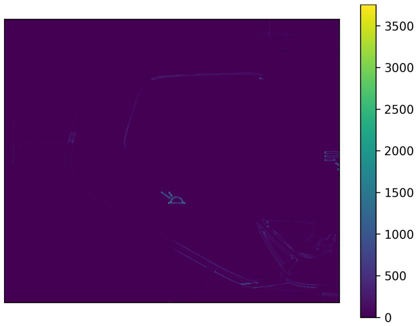
The rendering noise (cf. Sec. 3) is not detectable from a single image, because any rendered image is only an approximation but we would need a perfectly rendered scene against which we could compare. Therefore, we render the same scene multiple times (50 in this paper). It is important to set a random seed. As a result, the imperfections, i.e., the spatial noise, occurs in different pixels for each rendered image. By looking at the rendered images (cf. Fig. 3) as a temporal sequence, like in a video, the noise now appears as temporal noise between the rendered images. Fig. 5 depicts the variance along the temporal axis of the rendered images. As can be seen, the noise decreases significantly with higher SPP as well with the denoiser turned on.



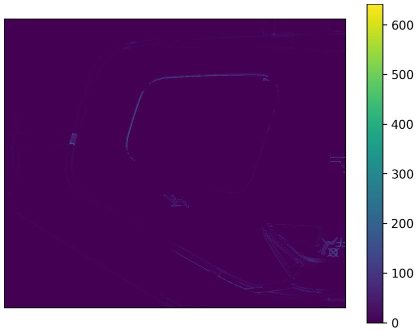
(a) 64 SPP



(b) 2500 SPP



(c) 64 SPP, denoised



(d) 2500 SPP, denoised

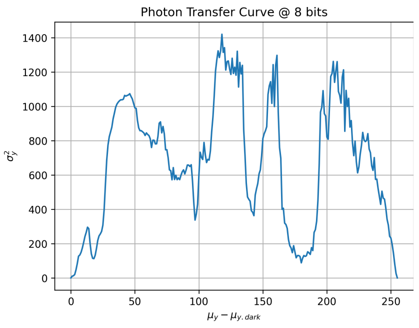
Figure 5: Variance along the temporal axes of the rendered images. Note the different scales.

5 Results

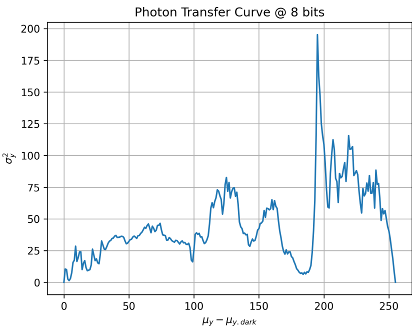
In this section we analyze how the rendering noise compares to the linear camera model.

To compute the photon transfer curve, we make use of the fact that the rendered data does not contain spatial non-uniformities as compared to a real camera sensor. Hence we do not have to apply the method described in the EMVA1288 standard and simply compute the average and variance along the temporal axis of the image sequence. We then quantize the average in bins with width one and average the variance at all pixels where the average has equal values.

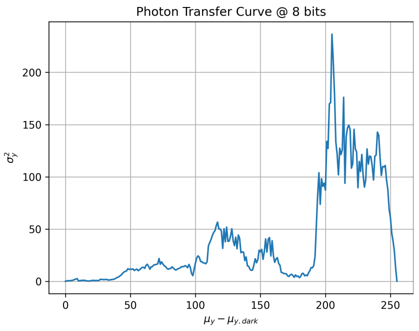
Finally we plot the variance against the average yielding the graphs as shown in Fig. 6. They are very noisy compared to the photon transfer



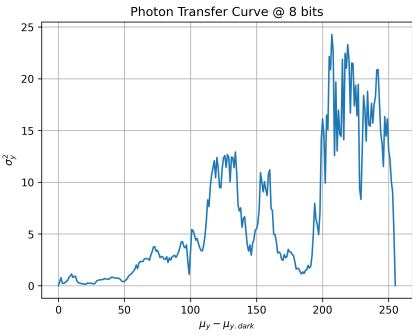
(a) 64 SPP



(b) 2500 SPP



(c) 64 SPP, denoised

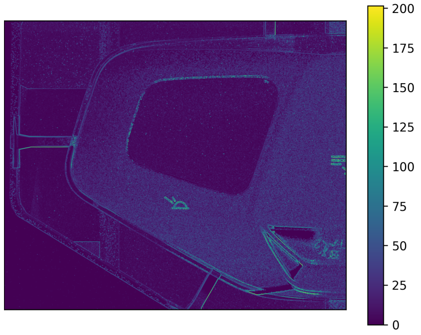


(d) 2500 SPP, denoised

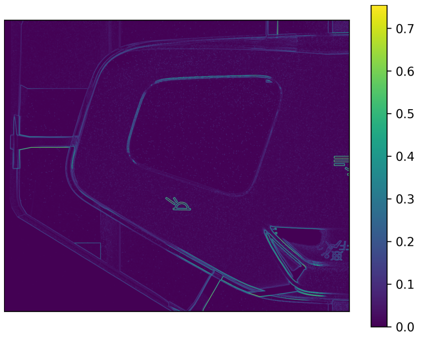
Figure 6: Photon transfer curve with different samples per pixel (SPP) and denoiser turned off or on.

curved of a real camera, cf. Fig. 2. They are non-linear and not even monotone. Therefore we conclude that the characteristic of rendering noise does not conform to the linear camera model according to the EMVA1288 standard.

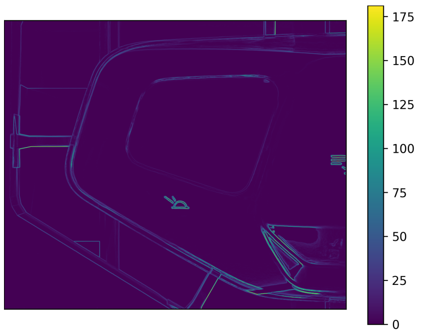
It is very insightful to note that the variance is high near edges, i.e. where there are strong brightness changes in the image. Fig. 7 depicts the Sobel-filtered images for the four different rendering settings; besides the scaling they look quite similar to Fig. 5. As expected, it appears that the noise is particularly high in regions with complex light propagation.



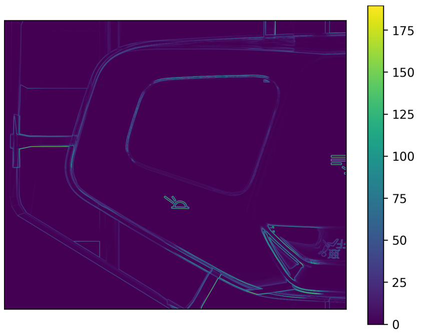
(a) 64 SPP



(b) 2500 SPP



(c) 64 SPP, denoised



(d) 2500 SPP, denoised

Figure 7: Sobel-filtered images with different samples per pixel (SPP) and denoiser turned off or on.

6 Proposed Method

A straight forward approach to minimize the noise is to render a scene multiple times and compute its variance along the temporal axis (cf. Fig. 5). We set for each pixel an individual SPP based on the targeted rendering time. It must be chosen in such a manner that the overall noise is minimal, i.e., low in regions with low temporal variance and vice versa. However, to render a scene multiple times (50 to 100) beforehand is extremely time consuming.

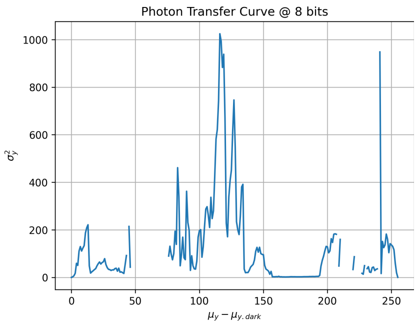
Therefore, we make use of the similarities between the variance images (cf. Fig. 5) and the Sobel images (cf. Fig. 7): The regions with strong edges can be extracted from a single rendered scene by edge detection, e.g. by using a Sobel-filter (cf. Fig. 7). This serves as an approximation for the variance but can be computed much faster.

If the photon transfer curves are now calculated without the regions with strong edges, the signal variance σ_y^2 is significantly reduced or even almost below 2. Here, the rendered image is practically noise free; to resemble the image of a real camera we can now add photon noise and dark current noise by simple parameterization based on the pixel gray values in compliance with the linear camera model [9].

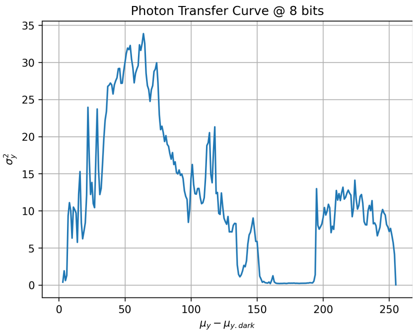
7 Summary

Synthetically generated data contains temporal noise that does not correspond to the photon noise of real cameras, but rather correlates with the complexity of the scene. The rendered image is less accurate and therefore more susceptible to noise in regions with edges or strong brightness transitions and in places that exhibit diffuse volume scattering.

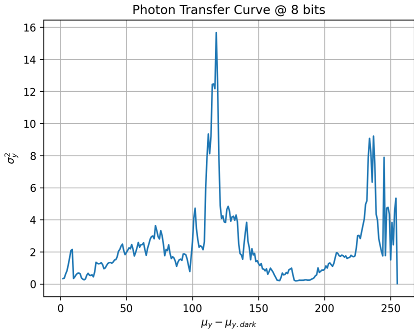
Based on these findings, a fast method was derived to identify the regions with high rendering noise from a single rendered image. These regions must be sampled with a higher SPP-setting, while the average SPP-setting is based on the targeted rendering time.



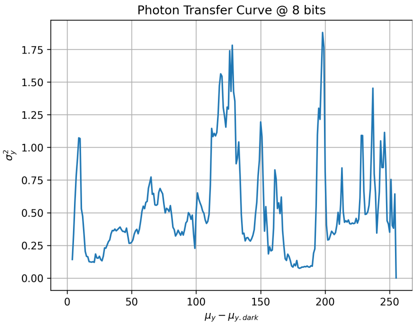
(a) 64 SPP



(b) 2500 SPP



(c) 64 SPP, denoised



(d) 2500 SPP, denoised

Figure 8: Photon transfer curve of filtered data with different samples per pixel (SPP) and denoiser turned off or on. The graph in (a) is discontinuous because some mean signal values μ_y no longer exist in the filtered data.

References

1. J. Meyer, T. Längle, and J. Beyerer, "About the Acquisition and Processing of Ray Deflection Histograms for Transparent Object Inspection," in *Irish Machine Vision & Image Processing Conference Proceedings*, 2016, pp. 9–16.
2. ——, "General Cramér-von Mises, a Helpful Ally for Transparent Object Inspection Using Deflection Maps?" in *Image Analysis*, P. Sharma and F. M. Bianchi, Eds. Cham: Springer International Publishing, 2017, vol. 10269, pp. 526–537, series Title: Lecture Notes in Computer Science. [Online]. Available: https://link.springer.com/10.1007/978-3-319-59126-1_44
3. J. Meyer, "Light Field Methods for the Visual Inspection of Transparent Objects," Ph.D. dissertation, 2019, medium: PDF Publisher: KIT Scientific Publishing. [Online]. Available: https://publikationen.bibliothek.kit.edu/1000091872
4. ——, "Next on stage: 'mc visi' – a machine vision simulation framework," Karlsruhe Institute of Technology, Tech. Rep. IES-2016-06, 2016.
5. J. Meyer, R. Gruna, T. Längle, and J. Beyerer, "Simulation of an inverse schlieren image acquisition system for inspecting transparent objects," in *Electronic Imaging*, 2016, pp. 1–9.
6. J. Meyer, T. Längle, and J. Beyerer, "About acquiring and processing light transport matrices for transparent object inspection," *tm-Technisches Messen*, pp. 731–738, 2016.
7. ——, "Acquisition and processing of light transport matrices for automated transparent object inspection," in *Forum Bildverarbeitung*, 2016, pp. 75–86.
8. ——, "Towards light transport matrix processing for transparent object inspection," in *Computing Conference*, 7 2017, pp. 244–248.
9. EMVA 1288 Working Group, "EMVA standard 1288 release 4.0 linear." [Online]. Available: https://www.emva.org/standards-technology/emva-1288/emva-standard-1288-downloads-2/
10. B. Jähne, *Digitale Bildverarbeitung und Bildgewinnung*, 7th ed. Springer Vieweg.
11. B. E. A. Saleh and M. C. Teich, *Fundamentals of Photonics*, 1st ed. Wiley. [Online]. Available: https://onlinelibrary.wiley.com/doi/book/10.1002/0471213748
12. J. R. Janesick, K. P. Klaasen, and T. Elliott, "Charge-coupled-device charge-collection efficiency and the photon-transfer technique," vol. 26, no. 10. [Online]. Available: http://opticalengineering.spiedigitallibrary.org/article.aspx?doi=10.1117/12.7974183

13. B. O. Community, *Blender - a 3D modelling and rendering package*, Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. [Online]. Available: http://www.blender.org

14. W. Jakob, S. Speierer, N. Roussel, M. Nimier-David, D. Vicini, T. Zeltner, B. Nicolet, M. Crespo, V. Leroy, and Z. Zhang, "Mitsuba 3 renderer," 2022, https://mitsuba-renderer.org.

15. A. T. Áfra, "Intel® Open Image Denoise," 2024, https://www.openimagedenoise.org.