

A Digital Image Processing Pipeline for Modelling of Realistic Noise in Synthetic Images

Oleksandra Bielova, Ronny Hänsch, Andreas Ley, Olaf Hellwich
Technische Universität Berlin

o.bielova@campus.tu-berlin.de, {r.haensch, andreas.ley, olaf.hellwich}@tu-berlin.de

Abstract

The evaluation of computer vision methods on synthetic images offers control over scene, object, and camera properties. The disadvantage is that synthetic data usually lack many of the effects of real cameras that pose the actual challenge to the methods under investigation. Among those, noise is one of the effects more difficult to simulate as it changes the signal at an early stage and is strongly influenced by the camera’s internal processing chain. The resulting noise is highly complex, intensity dependent, as well as spatially and spectrally correlated. We propose to transform synthetic images into the raw format of digital cameras, alter them with a physically motivated noise model, and then apply a processing chain that resembles a digital camera. Experiments show that the resulting noise exhibits a strong similarity to noise in real digital images, which further decreases the gap between synthesized images and real photographs.

1. Introduction

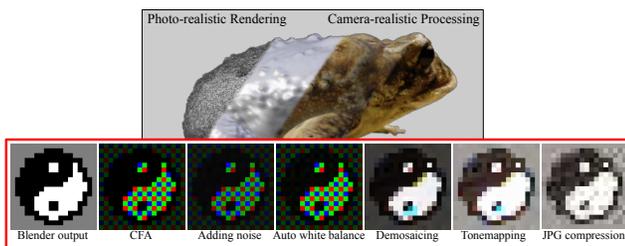


Figure 1. We transform noise-free synthetic images into the raw format of digital cameras, alter them with a physically motivated noise model, and apply a processing chain that resembles a real camera. This leads to images with very realistic statistical properties and increases the transferability of conclusions drawn from corresponding experiments.

Benchmark datasets are in high demand in areas such as autonomous driving [10], detection and tracking of ob-

jects [33], face recognition [22], bioimaging and biomedicine [30], optical flow, disparity estimation, 3D reconstruction [17, 9], and the training of neural networks [31].

These benchmark datasets provide not just the image data itself, but also reference data acquired by a sensor that is assumed to be significantly more accurate than the methods under investigation. As these datasets are based on real measurements, they potentially contain every aspect of the data acquisition process as it would be carried out during application of the analysed methods. Thus, the acquired evaluation results are assumed to have meaningful implications about the performance of methods when applied outside of the laboratory. The creation of such benchmark datasets, however, is time consuming and costly in particular with respect to the generation of the reference data. This causes a strong limitation on the number of possible variables that are covered, i.e. often only a single sensor (e.g. camera) with specific settings (e.g. depth of field, focal length, ISO, etc.) is used, scene and object properties such as lighting, surface texture, geometric complexity do not span the whole range of realistic variation, and the reference data is often far from problem-free (e.g. missing data, measurement errors or inaccuracies). While real data has usually more variation in scene and object properties than synthetic images, once a dataset is acquired its properties can’t be “modified” to test different usecases.

A solution to these issues is to use synthetic data which offers full control about scene, object, and sensor properties on the one hand and actual ground truth values for the target variables on the other hand. They can even be adapted to fulfill specific requirements as long as the basis of the synthesis (e.g. the corresponding Blender model) is still available. While synthetic benchmarks can not replace experiments on real world data, they offer complementary possibilities to evaluate certain aspects of a processing chain within a wide span of sensor settings and scene properties. The *Tsukuba* dataset [25, 24] consists of 1800 ground truth disparity maps and the corresponding image pairs with different lighting settings that had been created via Pixologic ZBrush and Autodesk Maya. The *Sintel* dataset [1] is cre-

ated from an animated 3D movie rendered with Blender, provides camera calibration and poses as well as the ground truth depth maps, and can be used to estimate performance on different tasks including depth and camera motion estimation. The creation of such synthetic datasets comes with its own challenges and limitations. On the one hand, it requires - besides a solid understanding of the involved rendering tools - a certain artistic skill which is why often simplified scenes are used or models originally intended for other purposes. On the other hand, there is an obvious gap between rendered images and images acquired with a real camera as the image synthesis often neglects effects of a real camera (such as tone mapping, lens distortions, noise, and motion blur). The work in [21] aims to decrease those discrepancies by synthesizing realistic images for a Synthetic Benchmark for 3D Reconstruction (*SyB3R*). This framework allows to investigate how calibration errors, camera parameters, and scene properties influence Structure from Motion and Multi-View Stereo. Blender and the path tracer Cycles as well as public 3D models are used to render photo-realistic images in a first step. Camera effects such as depth of field, scene properties such as illumination, and object properties such as surface texture are controlled within Blender. Other aspects such as tone mapping, radial distortions, and image noise are handled by a post processing chain to avoid the time consuming rendering process for every parameter change. One of the most difficult challenges is the creation of realistic image noise, which *SyB3R* models as a post processing step by adding a random noise term that is modelled after the statistical data of a real camera. The applied model, however, is rather simplistic and - despite creating visually consistent noise - cannot cover all statistical properties of noise in real cameras. Indeed, many approaches to model image noise focus on already interpolated RGB images [21, 11, 16] or even on JPEG files [26]. There are many noise sources affecting images taken with digital cameras. They may depend on different scene as well as sensor properties and may appear more or less prominent, but all of them (besides compression artifacts) are already present in the unprocessed output of the sensor and get shaped by subsequent processing operations [26, 3]. Thus, an alternative approach to adding the noise at the very end of the processing chain and aiming to shape the outcome to the statistical properties of a real camera, is to apply a realistic and physically motivated noise model at the very beginning of the image formation process and let the resulting image pass through a software equivalent of the processing pipeline of a real camera such as white balancing, demosaicing, tone mapping, gamma correction, etc. Such a solution provides not only the possibility to apply a realistic (physically motivated) noise model, but offers also optimal control about realistic image effects caused by the image processing chain of real cameras.

While every digital camera passes the acquired data through a processing chain, the specific steps are hard-wired inside the camera, proprietary, as well as confidential, and thus there are only general models (e.g. [23, 13, 14]) that often lack details specific to a given camera. A good overview of the color image processing pipeline of modern digital cameras can be found in [29], while a more high-level overview is provided in [3] which mainly focuses on processing steps that tend to create, enhance, or reduce different artifacts. Besides those abstract models of the in-camera processing pipeline, there exist few approaches that aim to implement at least certain aspects of this processing chain in software. In [16] a new in-camera imaging model is proposed based on the analysis of more than 10K images from more than 30 cameras. Their framework allows to convert a given sRGB image captured with one set of settings to an sRGB image with another set of settings of a specific camera. The pipeline includes a transformation of the sRGB image to the original RAW format, followed by white balancing, color space transform, gamut, and tone mapping. The proposed in-camera imaging model does not consider the demosaicing step and thus assumes that the raw values are already interpolated. The work in [15] provides a software platform that applies common camera imaging pipeline steps to images in RAW format. It allows the user to access each of these steps, to modify its parameters, to change intermediate images, and to re-introduce them into the pipeline.

We extend the work in [21] by using the framework of *SyB3R* to render photo-realistic and physically plausible images. These images are in a linear RGB color space that allows to transform them into mosaicked images as in the very beginning of the image formation process of a real camera. At this point we apply a physically motivated noise model that consists of multiple components, each of them adjustable by system parameters. Thus, noise is added to the synthetic images at a point where it would also be present in real cameras, namely before any image processing steps are executed. The created noisy RAW images are then fed to common processing steps as they would occur in a real camera, i.e. white balancing, demosaicing, noise reduction, tone mapping, gamma correction, and JPEG compression. The main contributions of our work are:

- An image processing pipeline that resembles the internal processing chain of real digital cameras. The proposed pipeline can be applied either to noise-free synthetic images or real images with high signal-to-noise ratio.
- We model synthetic image noise at the very beginning of the proposed pipeline where common assumptions about image noise (e.g. being IID) are still valid. The noise consists of three independent components:

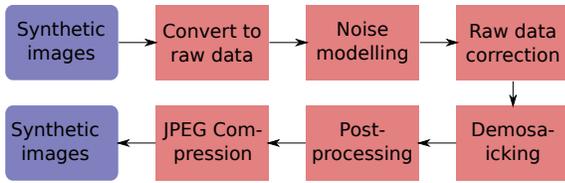


Figure 2. General overview of the proposed framework.

Poissonian-Gaussian noise, time-varying row noise, and fixed pattern noise. The introduced noise is transformed and shaped by the subsequent image processing steps very alike to the processes in real digital cameras.

- We show that the combination of the proposed noise-model and processing chain leads to results in the example task of multi-view stereo reconstruction that closely resemble results based on real images, where results with IID Gaussian noise lead to very different results.
- We provide a flexible, open-source C++ implementation [19] of the proposed framework that allows to easily change or extend individual modules, to modify parameter settings, to select different variants of a module or deselect it at all if it is not required (e.g. reverse auto white balancing is not necessary if the input images have not been white balanced and the denoising procedures may be skipped as well) and to have access to all intermediate images at each processing stage.

2. Methodology

Figure 2 shows the general workflow of the proposed method: As input serve synthetic (noise-free) images which are transformed into raw image data (Section 2.1) which are subsequently altered by different types of noise (Section 2.2). After applying the noise model, the images undergo a typical image processing pipeline (Section 2.3) consisting of auto white balancing, demosaicing, denoising, tone mapping, and compression.

2.1. Conversion into raw data

As input we use images rendered in Blender using Cycles which allows to take a multitude of scene, object, and camera properties into account. The proposed framework, however, is not limited to Blender and can be applied to any kind of images as long as they meet certain assumptions, i.e. having no or only weak noise and being in a linear color space. It is noteworthy that even real images can be used by the proposed processing chain as long as they have a large

signal-to-noise ratio¹ and are given in raw format (see Section 3 for examples).

As image sensors of digital cameras are not color-sensitive, a Color Filter Array (CFA) is used that lets only specific colors pass at each pixel. That is why the raw output of a sensor does not provide several color measurements per pixel, but every pixel measures the intensity of a different color. There are various CFA patterns and which one is used depends on camera vendor and model. The proposed framework uses RGB images and Bayer pattern (an example is shown in Figure 1), as they are widely used in digital cameras.

As the input images already provide RGB values for all pixels, they are converted to a CFA structure by using only one of those channels in each pixel. The remaining pixels are set to zero and are not used in further operations.

The output received from the image sensor of a camera usually has incorrect white balance (WB). If the input image is already white balanced, we revert this step by dividing red and blue values by user-defined coefficients.

2.2. Noise modelling

The noise in the final image product of a digital camera is highly complex, intensity dependent, and correlated spatially as well as over the different color channels. The core idea of the proposed framework is not to attempt to model this highly complex noise of the final image product as it was done e.g. in [21], but instead model the noise at the very beginning of the processing pipeline, where the corresponding model is considerably simpler. Nevertheless, even at this point the exact noise model is not completely known and target of scientific interest [27]. We decided to use the Poissonian-Gaussian noise model [5] as it is widely assumed as being suitable for the raw data of digital image sensors on the one hand and relatively simple on the other hand. This is of importance since the user of the framework needs to be able to adjust its parameters efficiently to obtain realistic results.

The model consists of two independent components: The Poissonian component relates to the signal-dependent noise, such as photon shot noise, which follows the Poisson distribution due to the photon-counting process. The Gaussian component represents the remaining signal-independent noise, e.g. electric or thermal noise. The variance σ^2 of the measured signal can be described by

$$\sigma^2(y(x)) = ay(x) + b \quad (1)$$

$$a = \chi^{-1}\theta \quad (2)$$

$$b = \theta^2 b' + b'' - \theta^2 \chi^{-1} p_0 \quad (3)$$

¹While in principle the framework is of course applicable to images of any noise level, it is unclear how noise sources additional to those being modelled would influence the final image noise.

where $\sigma_y(x)$ is the variance of Poissonian component, which depends on the original noise-free signal $y(x)$, χ is a real scalar parameter related to the quantum efficiency of the sensor (i.e. the more photons are necessary to generate a sensor response, the smaller is χ), $\theta > 1$ is a scaling factor corresponding to the signal amplification (i.e. a analog gain to amplify the collected charge usually controlled by setting the ISO sensitivity in digital cameras). The constant variance b of the Gaussian component has two components: b' and b'' , where the latter describes the part of the noise that appears after amplification and is not affected by θ . The pedestal parameter p_0 is the base level of the image sensor to which the collected charge is added.

Besides the Poissonian-Gaussian noise, we additionally model row noise and vertical (column) fixed pattern noise (VFPN) [11]. These noise types follow a Gaussian distribution and can be simply generated by adding Gaussian distributed random values to each row (or column), respectively. The difference between both is that row noise is time-varying and is thus different among different frames, while VFPN remains immutable over multiple frames and differs only in images taken with different camera models.

2.3. In-camera image formation

Intensity scaling. Intensity values synthesized by SyB3R based on Blender are in a linear color space and range between 0 and 1. While the raw data of digital cameras is often also linear in intensity, the range varies between different camera models. Many cameras specify the black level of the sensor, i.e. the minimum value which usually deviates from zero, as well as the white level. If no black and white levels are specified, the proposed framework uses the minimum and maximum values of the image for normalization.

Auto white balancing. Automatic white balancing (AWB) is used in digital cameras to achieve color constancy, i.e. attempting that the colors of an object do not vary for different illuminations by adjusting signal levels such that the color spectral response of the image sensor is similar to the human eye [14].

While there are many types of AWB (see e.g. [34] for an overview of basic techniques for digital cameras), the two most widespread methods are “Gray world” (based on the assumption that the mean scene reflectance is achromatic) and “White patch” (based on color constancy, i.e. the maximal cone response of the human visual system is perceived as white) [18].

Both methods lead to different results in practice (the white patch approach provides “warmer” images that are usually more pleasant to the human eye), which is why the proposed framework implements both and leaves it to the user to select the one better suited to his needs.

Demosaicing. Demosaicing is an integral part of every image formation process in single-sensor digital cameras and is necessary for full-color image reproduction that accurately represents the captured scene [29]. There are several well-established demosaicing algorithms of which [28] provides a good overview. While the most efficient demosaicing approach is a simple non adaptive bilinear interpolation, better results are obtained by edge-directed interpolation techniques that analyze the local pixel neighborhood to determine a preferred interpolation direction and thus avoid interpolating across image edges [12]. Median-based interpolation [6] combines bilinear interpolation and median filtering of color differences and has been shown to produce superior results for real images [28]. The proposed framework provides all three of the above variants and proposes an additional fourth method, namely edge-directed median-based interpolation, that combines edge-directed interpolation and median-based filtering.

Figure 3 shows example results of these demosaicing techniques applied to a noisy CFA image obtained from a camera (a Canon 5D Mark II). The two methods based on median filtering (Figure 3c) and d)) preserve more sharpness. Bilinear as well as edge-directed interpolation (shown in Figure 3a) and b)) lead to color artifacts. Bilinear interpolation causes color fringes at edges (Figure 3a) which are also not corrected if median filtering (Figure 3c) is applied subsequently. At least for this example image, the result of the proposed method (Figure 3d) appears most similar to the output of the camera (Figure 3e)).

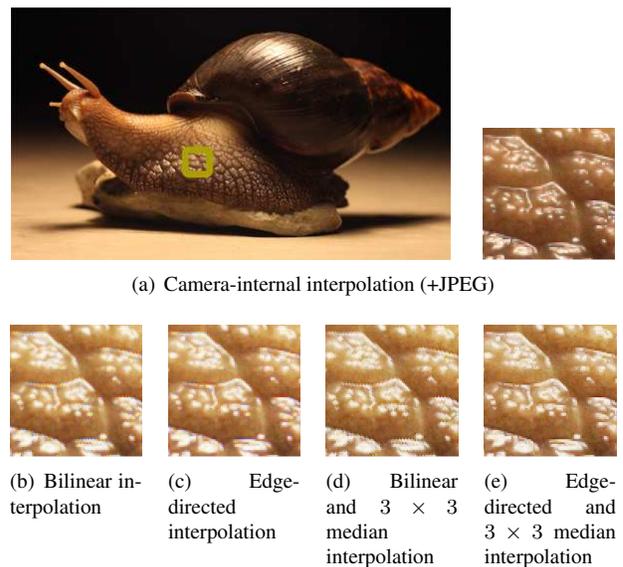


Figure 3. Different interpolation techniques applied to a real image in RAW format.

Denoising. Most modern cameras internally apply a denoising method before the non-linear tone-mapping stage of which the proposed framework provides three simple examples: A simple *average filter*, a *median filter* that preserves edges but effectively only removes outliers, and a *bilateral filter* that preserves edges by considering both pixel position and intensity. Denoising is performed in YCbCr color space with a stronger filtering on the chroma components. The amount of filtering can be steered by the ISO level.

Tone-mapping. Tone-mapping is part of the photo-finishing process and aims to simulate the appearance of a scene with a high-dynamic range in a medium with limited range. Its non-linear transformation is a rather artistic choice, usually differs for different camera models as well as settings, and is implemented as 1D Look-Up Table (LUT). If no tone curve is specified by the user, the proposed framework uses the tone curve from [15] as default.

Gamma correction. Since the human eye perceives differences in dark regions much better than the differences in light regions, it is desirable to have a larger accuracy for low intensities, while high intensities could be compressed to save space. This task is solved by Gamma correction [4, 2] that transforms every intensity value by the power function

$$f(x) = x^{1/\gamma} \quad (4)$$

that describes the relationship between the numerical value of a pixel and its actual luminance. Cameras usually implement Gamma correction as an 1D LUT. The proposed framework uses as default $\gamma = 2.2$ as an approximation of sRGB.

Image compression. Although some digital cameras allow to store images in raw format, it is more common to compress the final image product to be able to save more images on a single memory card. JPEG is commonly used for (lossy) compression of digital images as it allows to specify the compression ratio with the cost of compression artifacts (e.g. colorful halos around edges) if too strongly compressed. Before JPEG compression, pixel values are converted to an 8-bit representation, i.e. to values in the range [0–255]. The proposed framework uses a default compression setting of 97.

3. Experiments

The purpose of the proposed framework is to bring synthetic images closer to their real-world counterparts in order to increase the transferability of conclusions drawn from experiments on synthetic data to the application on real-world data. To this aim, care was taken to apply a physically plausible and well established noise model at the very beginning of a processing chain which resembles the image formation pipeline in a digital camera. It is, however, not without difficulty to attempt to evaluate this procedure. The statistics

and properties of a captured image vary greatly being not only dependent on scene properties, but also on the camera model (i.e. the specifically implemented processing chain) and its settings (e.g. tone-mapping, compression ratio, etc.). Nevertheless, we attempt to qualitatively show that the proposed framework leads to similar noise characteristics as in images captured with a real camera (Section 3.1) and to provide quantitative cues that the produced synthetic images lead to similar behaviour of methods in an example application (Section 3.2).

3.1. Qualitative evaluation

As an example we use an synthetic image of a toad as natural object with strong texture as well as an image of a snail as its real counterpart captured by a camera (i.e. a Canon 5D Mark II). As far as possible, the same parameter settings are used for capturing the photos and synthesizing the images: Focal length of 50.0 mm, an aperture of 4.0, and the sensor width is 36 mm with a resolution of 1920×2880 pixels. We used a ISO level of 3200 to make noise well visible.

The images are simultaneously stored in JPEG and CR2 (raw) format which allows to apply the image processing pipeline of the proposed framework to real, noisy, and unprocessed raw data of actual photographs. As those images already contain (real) image noise, the noise-generation step of the pipeline is omitted.

While Figure 4(b) shows the processing results of the proposed framework when applied to a synthetic image, Figure 4(d) shows a real image that is processed with the identical pipeline (besides the noise modelling step). In both cases noise appears intensity dependent and spatially correlated. Figure 4(e) shows the results of the camera-internal processing chain and illustrates clearly the influence of the specific pipeline on the final image characteristics: In both cases, i.e. Figures 4(d) and 4(e), the exact same image is used including the noise (as it is the same picture). Only the image processing pipeline differs and leads to a different coloring (due to tonemapping) and seemingly slightly different noise (e.g. due to different noise-reduction). However, this variation only reflects the inter-camera variance regarding specific details of the applied processing chain.

For reference, Figures 4(a) and 4(c) show the results of [21] which models image characteristics after measurements taken for a specific camera (in this case a Canon EOS 400D). Besides applying a different tonemapping, this camera has also more noise at this ISO level. Furthermore, the noise model is applied as a post processing step and thus not influenced by previous processing steps.



(a) Synthetic image; Noise and image processing from [21].



(b) Synthetic image; Proposed noise model and image processing.



(c) Real image; Real noise and image processing from [21].



(d) Real image; Real noise and proposed image processing.



(e) Real image; Real noise and in-camera image processing.

Figure 4. Qualitative comparison between synthetic noise generated by [21] and the proposed model on synthetic and real images. Differences best seen in the digital version.

3.2. Quantitative evaluation

In this section, we consider the application of multi-view stereo (MVS) 3D reconstructions where, especially on weakly textured surfaces, noise plays a fundamental role [21]. We compare five types of images: The unaltered out-of-camera JPEG images, the raw images with camera noise but processed with our pipeline, “noise free” raw images processed with our pipeline including synthetic noise, “noise free” raw images processed with SyB3R [21], and JPEG images with Gaussian IID noise with the same variance as the out-of-camera JPEG images. For four of these types, we create images with increasing noise levels and

analyze, how the point clouds from the 3D reconstruction deteriorate. SyB3R uses a noise model with settings that fit a specific camera with specific parameters and does - in contrast to the proposed work - not allow a free choice of the ISO level. Thus, there are only results for ISO 1600.

To rule out any influence of a synthetic image creation, we base this experiment on real photos. We took pictures with a Sony A7R II of a scene containing strongly and weakly textured surfaces (see Figure 5). The scene is pictured from six different view points to enable 3D reconstruction, which was carried out with a custom structure from motion pipeline followed by PMVS2 [8] for dense MVS reconstruction. For each view point, we captured images for the ISO levels 100, 200, 400, 800, 1600 while compensating for increased brightness with the exposure time. For each view point and ISO level, eight images were taken (i.e. $6 \times 5 \times 8$ images in total).

The first two types of images are procured by simply taking, for each ISO level and view point, one of the eight images and using either the JPEG or by processing the raw file with our pipeline, respectively. The third type, however, requires noise free raw data as well as a calibrated synthetic noise model. For each view point, we average the eight ISO-100 images (in their raw form) to compute an “ISO-12.5” image which we consider to be noise-free (similar to [20]). To calibrate the synthetic noise, we compute for each ISO level the pixel variations across the eight images of each view point. By combining the estimated variance for each pixel with the estimated true intensity from the “ISO-12.5” images, we can perform a least squares fit of the parameters a and b in Equation (1) for each ISO level. We then run the proposed processing chain on the “ISO-12.5” images with the estimated parameters for the synthetic noise. To create the fourth image type, JPEG images with IID noise, we compute the pixel variance in the out-of-camera JPEG images for each ISO level and add Gaussian IID noise with this variance to the averaged ISO 100 out-of-camera JPEG images.

Figure 5 shows crops from the five image types. Note that the real and our proposed synthetic noise are spatially correlated. Also note how for the real and proposed processing, with increasing ISO level, not only the noise increases but also, due to increased filter strength (as being dependent on the ISO level, see Section 2.3), the texture is more and more removed.

For each of the five processing types and for each ISO level, we perform a 3D reconstruction. For two different confidence thresholds in the dense reconstruction, Figure 6 shows the number of points in the resulting point clouds which usually correlates with a reconstruction’s completeness. As expected, the ability of PMVS to locate 3D points decreases with increasing noise. The numbers of points for the out-of-camera JPEGs and the images produced by the

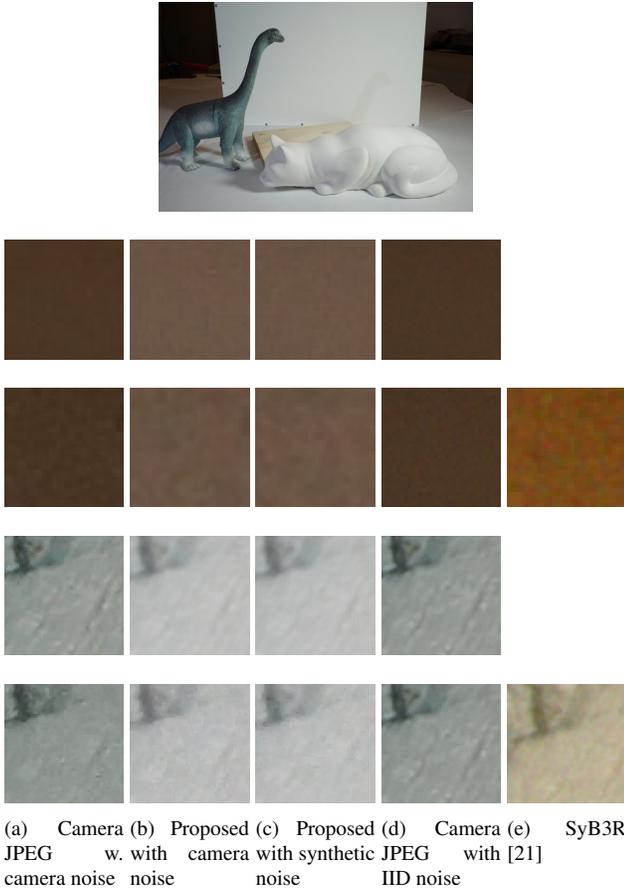


Figure 5. Example image (top row) and crops for different types and ISO levels. Color differences are caused by different tone mapping procedures, i.e. proposed (2nd and 3rd column) and camera internal (1st and 4th column). Rows alternate between ISO 100 and ISO 1600. Differences best seen in the digital version.

proposed pipeline are quite similar. The IID noise, however, has a completely different impact on the reconstruction. This is due to two reasons: Firstly, PMVS by default downsamples the input images internally to half the resolution. This is not uncommon, other frameworks (e.g., MVE [7]) do the same, and even the Middlebury 2014 images [32] are not full resolution images. Downsampling IID noise reduces much of the noise energy (all the high frequency components). Real noise, on the other hand, is spatially correlated and thus does not lose as much energy from downsampling. The second reason for the different behavior is that the IID noise only reflects the noise strength of the real images, but not the degradation of the signal due to stronger filtering for higher ISO levels. Using IID noise, even if calibrated to have the correct strength, thus leads to an overestimation of a method’s resilience to noise. The noise model in SyB3R [21] needs to be fitted to each camera ISO level individually and was only available for ISO 1600. There, it

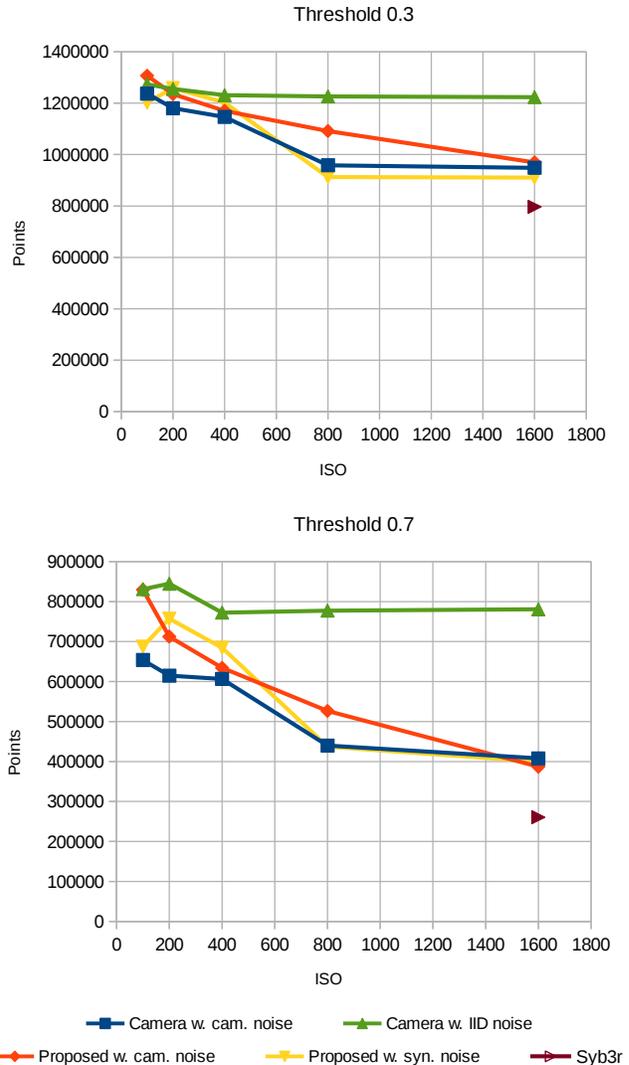


Figure 6. Number of points in the reconstructed point clouds for different processing types and noise levels. SyB3R refers to the noise model in [21] which was tweaked for a different camera (Canon EOS 400D) that has more noise at ISO 1600 than the Sony A7R II used and imitated here.

seems to behave more realistic than IID noise, but is not as close as the proposed noise model. However, the comparison is not entirely fair since the noise model in SyB3R is modeled after a different camera (Canon EOS 400D) which also has more noise at the same ISO level in reality. Thus, a less complete reconstruction at the same ISO level is to be expected.

To judge the precision of the reconstruction, we performed Poisson surface reconstructions. These can be found in Figure 7 for three different ISO levels. The images are shaded by ambient occlusion to better highlight inaccuracies. As with completeness, precision is also af-



Figure 7. Meshed 3D reconstructions with a low threshold (0.3). Top to bottom: Camera w. camera noise, proposed w. camera noise, proposed with synthetic noise, camera w. IID noise. Left to right: ISO 100, 400, 1600.

ected by image noise and deteriorates quickly for weakly textured surfaces. The precision for images processed with the proposed approach seems to be very close to the precision achieved with the out-of-camera JPEGs albeit slightly better. We attribute this difference to the difference in tonemapping and filtering which varies for different real world cameras. With IID noise, the precision seems to decrease only very slightly and the results are very far from the real world.

4. Conclusions and Future Work

The proposed framework models realistic noise and other effects in synthetic images by simulating the image formation process of digital cameras. The framework operates with HDR images that can for example be synthesized in Blender but can also be applied to real camera images if they are provided in RAW format. The case of synthetic images, however, has the advantage of allowing control over camera, scene, and object parameters on the one hand, and on the other providing access to ground truth values for target variables such as depth, albedo, etc. The intended application is the creation of synthetic benchmark datasets for the evaluation of image based algorithms. Unlike other works, the noise in the final image is not modeled based on data from one particular camera model, but simulated at the very beginning of the image formation process, i.e. in the raw data, before any processing operations are applied. While the noise model is simple, it considers the most significant, signal-dependent and signal-independent noise sources. The simplicity of the model allows control

over noise type and energy by a few parameters with a concise physical interpretation such as parameters related to the ISO level and the quantum efficiency of the sensor.

The framework implements all main processing steps inherent to an in-camera imaging pipeline, i.e. intensity scaling, auto white balancing, demosaicing, denoising, tone mapping, gamma correction, and compression. All intermediate results of each stage are accessible and can be stored as HDR files. The implementation of the proposed image processing pipeline is flexible and allows the user to select which steps should be performed with which parameter settings.

Qualitative results show that the synthetic noise closely resembles realistic camera noise. The main differences are caused by different image processing pipelines of the proposed framework and the used example camera, in particular with respect to different tone mapping and denoising techniques. As the specific processing chains also differ significantly for different camera vendors or even image settings and the proposed framework is flexible regarding the individual modules, the resulting images are well within the realistic range. Quantitative results on the example task of MVS show, that the corresponding methods behave very similar if either real images or images with the proposed synthetic noise are used. Images with Gaussian IID noise, however, lead to very different results demonstrating clearly the insufficiency of this simple noise model.

There are several aspects that require a more elaborate implementation to obtain even more accurate results. While the proposed noise model appears to be sufficiently accurate, the processing chain can be improved by using a more sophisticated implementation of individual modules. One example is the denoising step, where the proposed pipeline applies rather simple methods while modern cameras seem to use approaches that filter noise more effectively while preserving fine image structures. Furthermore, while several works regarding camera pipelines (e.g. [29, 15]) perform denoising after the demosaicing step, it might be beneficial to apply it before any processing steps alter the nature of the noise too much. Finally, most cameras do not directly record the intensities of the sRGB primaries, something the camera has to compensate for. This color space transformation was neglected in this work and could be added in the future.

References

- [1] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), editor, *European Conf. on Computer Vision (ECCV)*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, 2012.
- [2] Y. Cao, P. Xiaofang, X. Zhao, and H. Wu. An analog gamma correction scheme for high dynamic range cmos logarithmic

- mic image sensors. *Sensors (Basel, Switzerland)*, 14:24132–24145, 12 2014.
- [3] A. Deever, M. Kumar, and B. Pillman. Digital camera image formation: Processing and storage. In *Digital Image Forensics*, pages 45–77. Springer, New York, NY, 11 2013.
- [4] M. Farshbaf Doustar and H. Hassanpour. A locally-adaptive approach for image gamma correction. In *10th International Conference on Information Sciences, Signal Processing and their Applications, ISSPA 2010*, pages 73–76, 06 2010.
- [5] A. Foi, M. Trimeche, V. Katkovnik, and K. Egiazarian. Practical poissonian-gaussian noise modeling and fitting for single-image raw-data. *IEEE Transactions on Image Processing: a publication of the IEEE Signal Processing Society*, 17:1737–1754, 11 2008.
- [6] W. T. Freeman. Median filter for reconstructing missing color samples, 02 1988. US Patent 4,724,395.
- [7] S. Fuhrmann, F. Langguth, and M. Goesele. Mve: A multi-view reconstruction environment. In *Proceedings of the Eurographics Workshop on Graphics and Cultural Heritage, GCH '14*, pages 11–18, Aire-la-Ville, Switzerland, Switzerland, 2014. Eurographics Association.
- [8] Y. Furukawa and J. Ponce. Accurate, dense, and robust multi-view stereopsis. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376, 2010.
- [9] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.
- [10] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361, 05 2012.
- [11] R. D. Gow, D. Renshaw, K. Findlater, L. Grant, S. J. McLeod, J. Hart, and R. Nicol. A comprehensive tool for modeling cmos image-sensor-noise performance. *IEEE Transactions on Electron Devices*, 54:1321–1329, 07 2007.
- [12] B. K. Gunturk, J. Glotzbach, Y. Altunbasak, R. W. Schafer, and R. M. Mersereau. Demosaicking: color filter array interpolation. *IEEE Signal Processing Magazine*, 22(1):44–54, 2005.
- [13] F. Heide, M. Steinberger, Y.-T. Tsai, M. Rouf, D. Pajak, D. Reddy, O. Gallo, J. L. abd Wolfgang Heidrich, K. Egiazarian, J. Kautz, and K. Pulli. Flexisp: A flexible camera image processing framework. *ACM Transactions on Graphics (Proceedings SIGGRAPH Asia 2014)*, 33(6), 12 2014.
- [14] W.-C. Kao, S.-H. Wang, L.-Y. Chen, and S.-Y. Lin. Design considerations of color image processing pipeline for digital cameras. *IEEE Transactions on Consumer Electronics*, 52:1144–1152, 12 2006.
- [15] H. C. Karaimer and M. S. Brown. A software platform for manipulating the camera imaging pipeline. In *Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science*, volume 9905, pages 429–444. Springer, Cham, 10 2016.
- [16] S. J. Kim, H. Ting Lin, Z. Lu, S. Susstrunk, S. Lin, and M. S. Brown. A new in-camera imaging model for color computer vision and its application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34:2289–2302, 02 2012.
- [17] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, 36(4), 2017.
- [18] E. Lam and G. S. K. Fung. Automatic white balancing in digital photography. *Single-Sensor Imaging: Methods and Applications for Digital Cameras*, pages 267–294, 09 2008.
- [19] A. Ley, R. Hänsch, and O. Hellwich. Project Website. <http://andreas-ley.com/projects/SyB3R/>.
- [20] A. Ley, R. Hänsch, and O. Hellwich. Reconstructing white walls: Multi-view, multi-shot 3d reconstruction of textureless surfaces. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, III-3:91–98, 2016.
- [21] A. Ley, R. Hänsch, and O. Hellwich. Syb3r: A realistic synthetic benchmark for 3d reconstruction from images. *Computer Vision – ECCV 2016*, 7:236–251, 10 2016.
- [22] H. Liu, H. Duan, H. Cui, and Y. Yin. Face recognition using training data with artificial occlusions. In *016 Visual Communications and Image Processing (VCIP)*, pages 1–4, 11 2016.
- [23] H. Liu and F. Yu. Research and implementation of color image processing pipeline based on fpga. In *Conference: 2016 9th International Symposium on Computational Intelligence and Design*, pages 372–375, 12 2016.
- [24] M. P. Martorell, A. Maki, S. Martull, Y. Ohkawa, and K. Fukui. Towards a simulation driven stereo vision system. In *ICPR2012*, pages 1038–1042, 2012.
- [25] S. Martull, M. P. Martorell, and K. Fukui. Realistic cg stereo image dataset with ground truth disparity maps. In *ICPR2012 workshop TrakMark2012*, pages 40–42, 2012.
- [26] S. Nam, Y. Hwang, Y. Matsushita, and S. Joo Kim. A holistic approach to cross-channel image noise modeling and its application to image denoising. In *Conference: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1683–1691, 06 2016.
- [27] T. Qiao, F. Reirant, R. Cogranne, and T. Hai Thai. Source camera device identification based on raw images. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 3812–3816, 09 2015.
- [28] R. Ramanath, W. Snyder, G. L. Bilbro, and W. A. Sander III. Demosaicking methods for bayer color arrays. *J. Electronic Imaging*, 11:306–315, 07 2002.
- [29] R. Ramanath, W. Snyder, Y. Yoo, and M. Drew. Color image processing pipeline. *IEEE Signal Processing Magazine*, 22:34–43, 02 2005.
- [30] D. Sage, H. Kirshner, C. Vonesch, S. Lefkimmiatis, and M. Unser. Benchmarking image-processing algorithms for biomedicine: Reference datasets and perspectives. In *Proceedings of the 21st European Signal Processing Conference (EUSIPCO)*, 09 2013.
- [31] A. Saxena, S. H. Chung, and A. Y. Ng. 3-d depth reconstruction from a single still image. *International Journal of Computer Vision*, 76:53–69, 01 2008.
- [32] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nesić, X. Wang, and P. Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In X. Jiang, J. Hornegger, and R. Koch, editors, *GCPR*, volume 8753 of

Lecture Notes in Computer Science, pages 31–42. Springer, 2014.

- [33] B.-S. Shin, X. Mou, W. Mou, and H. Wang. Vision-based navigation of an unmanned surface vehicle with object detection and tracking abilities. *Machine Vision and Applications*, pages 1–18, 10 2017.
- [34] G. Zapryanov, D. Ivanova, and I. Nikolova. Automatic white balance algorithms for digital still cameras – a comparative study. *Information Technologies and Control*, 1:16–22, 01 2012.