Fully synthetic training for image restoration tasks

Raphaël Achddou, Yann Gousseau, Saïd Ladjal

Fully synthetic training for image restoration tasks

Raphaël Achddou[1],Yann Gousseau[1],Saïd Ladjal[1]

LTCI Telecom Paris Institut Polytechnique de Paris, 19 place Marguerite Perey, Palaiseau 91440, France
raphael.achddou@telecom-paris.fr

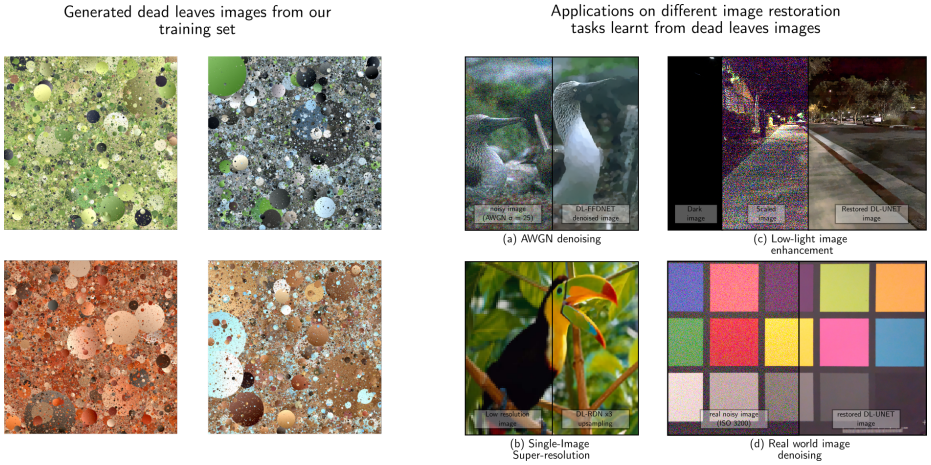
Abstract. In this work, we show that neural networks aimed at solving various image restoration tasks can be successfully trained on fully synthetic data. In order to do so, we rely on a generative model of images, the scaling dead leaves model, which is obtained by superimposing disks whose size distribution is scale-invariant. Pairs of clean and corrupted synthetic images can then be obtained by a careful simulation of the degradation process. We show on various restoration tasks that such a synthetic training yields results that are only slightly inferior to those obtained when the training is performed on large natural image databases. This implies that, for restoration tasks, the geometric contents of natural images can be nailed down to only a simple generative model and a few parameters. This prior can then be used to train neural networks for specific modality, without having to rely on demanding campaigns of natural images acquisition. We demonstrate the feasibility of this approach on difficult restoration tasks, including the denoising of smartphone RAW images and the full development of low-light images.

Keywords: Image image restoration · image denoising · statistical modelling

1 Introduction

Before the advent of deep neural methods for image restoration tasks, most approaches relied on relatively lightweight and explicit image priors. For instance, the use of total variation [50] as a regularization term is a consequence of a Laplacian prior on image gradients. Methods involving wavelet shrinkage [19] assume a regularity prior on wavelet coefficients related to Besov spaces. Non-local methods [8] rely on an auto-similarity hypothesis.

More recently, deep neural networks have achieved impressive results in all fields of image restoration: denoising, single image super-resolution, deconvolution, etc. In order to achieve such results, networks need to be trained on voluminous image databases. The resulting trained networks can be interpreted as image priors, even though it has been shown that the mere structure of networks can already be considered a prior [56]. In any case, such priors are non-explicit and involve a huge number of parameters. They also need to be retrained for each new acquisition conditions or specific imaging device [14].



(a) AWGN denoising (c) Low-light image enhancement

(b) Single-Image Super-resolution (d) Real world image denoising

Fig. 1: Left : Dead leaves images generated with our algorithm. Right : some results for different image restoration tasks, using neural networks trained on dead leaves images (single-image super-resolution, AWGN denoising, Low-light enhancement)

In this paper, we show that trainings on large image databases can be efficiently replaced by trainings on synthetic images. To achieve this, we rely on a mathematical model that is physically grounded and depends only on a few parameters, the scaling dead leaves model [4, 25, 35], which combines an occlusion-based dead leaves model with a scaling size distribution for objects. This model offers a good balance between simplicity and accuracy in accounting for natural images statistics. Preliminary results that we have presented in [2] show that this model has the potential to restore images corrupted with synthetic additive noise. To the best of our knowledge, this was the first work to propose a synthetic training for image restoration tasks. In the present paper, we extend and confirm these preliminary results by addressing difficult real world situations, including the denoising of RAW smartphone images and the full image development pipeline for extreme low light images. The strategy here is to train neural networks with databases that are fully synthetic, in the sense that both the clean images and the physical degradations are simulated. Surprisingly, we show that using generative models with only a very limited number of parameters is enough to produce near state-of-the-art performances, or even to outperform them in some cases where the validity of the ground truth is questionable. This implies that, for restoration tasks, the geometric contents of natural images can be nailed down to only a simple generative model and a few parameters. Our approach does not need real images ground truth and therefore bypass the need of costly image acquisition campaigns. Eventually, we show that our training yields a better restoration of the synthetic images used (scaling dead leaves). While this is not surprising per se, this facts confirms a better preservation of

some fine details, as can also be directly observed on some examples. Indeed, such synthetic images are the basis for a recent ISO norm evaluation of the respect of textures in images [29].

We believe that such a study both sheds light on the way convolutional neural networks can address restoration problems and opens interesting perspectives. First, this result shows that the mere structure of such networks is adapted to image restoration tasks and that despite their huge number of parameters they can be made near-optimal from just a few principles and hyper-parameters. This result, and the fact that simpler, less structured models cannot achieve satisfying restoration performance, also highlights the type of geometric structures a neural networks needs to be efficiently trained. Second, the proposed learning database can be modified according to specific acquisition devices and in particular to their point spread function, dynamic range, noise modality, etc. This yields flexible, generic and relatively light learning schemes, as illustrated for two real-world image restoration tasks (Smartphone RAW denoising and extreme low-light RAW image development).

Our paper is organized as follows : after presenting related works in image modelling and image restoration, we define the dead leaves model and explain how it is used to generate synthetic databases in Section 3. We then introduce in Section 4 the various image restoration tasks we studied. For each task, we explain how to adapt the generation schemes to the problem at hand and detail the degradation model that is used. Finally, we present our experimental results in Section 5.

2 Related Works

2.1 Image restoration priors

Classical Bayesian image restoration methods assume some closed form statistical prior on the images distribution. In turn, classical variational methods can be reformulated in this context. Among the models that fall in this general framework let us cite the Wiener model, for which the distribution of images is assumed to be Gaussian and translation invariant and the total variation [50] for which the log-likelihood is the l^1 norm of the gradient vector field. In [19] the authors derive an algorithm for restoring signals under the assumption that the targeted signals are well approximated by a sparse representation in some wavelet decomposition. Later, a consequent body of literature discussed the implications of the sparsity assumption and a variety of algorithms were proposed to take advantage of this particular form of regularity ([11, 18]). The total variation model itself can be viewed as a form of sparsity.

Self-similarity is another powerful hypothesis on the image distribution. Non-local algorithms, which are based on this assumption, leverage the redundancy of the content in natural images. Methods such as Non-Local Means [8], Non Local Bayes [34], or BM3D [17], average stacks of similar patches in order to denoise them in a collaborative way.

Deep learning methods for restoration seek directly to build a machinery (the trained network) that minimises a reconstruction error. Typically, the loss function is taken to be the mean square error between the perfect image and the output of the network. Here the prior is represented by the training dataset which is believed to convey sufficient information about the distribution of images, see e.g. [62, 63] among numerous other works. Recently, a great effort has been dedicated to create hybrid methods exploiting both the expressive power of convolutional neural networks and the self-similarity assumption. These methods showed impressive performances on a variety of computer vision tasks [57], including image denoising [16, 36].

In turn, a trained denoising network can be used as an implicit prior on the image distribution. This idea, named plug and play, consists in using the denoising network in place of a proximal operator during an iterative optimisation of a variational model [43]. Another approach considering a network as a prior on images is presented in [56] in which it is showed that the architecture itself of a network can serve as a regulariser.

2.2 Deep Learning for image restoration in real world conditions

Over the last few years, methods based on deep learning have set new standards for most image restoration tasks. In order to restore images properly, neural networks are usually trained with pairs of distorted and clean images. Creating such pairs is straightforward as long as we have a suitable degradation model at hand. For example, noise can be modelled as an additive white gaussian noise (AWGN) [63] and for super-resolution, it can be assumed that a bicubic downsampling is applied to a full resolution image [65]. Neural networks trained in such a fashion produce outstanding results when confronted with degradations they have been trained on, but fail to generalize to real-world cases [66]. This is problematic, especially when the deterioration models are over-simplified, as it is e.g. the case for the two aforementioned examples.

A first approach to deal with the complex image degradations that occur in real acquisition conditions is to directly collect pairs of clean and degraded images, therefore by-passing the modeling of the degradations. For example, the See-in-the-Dark dataset [14] is composed of thousands of images taken in low-light conditions with varying exposure time. Training pairs are formed with short-exposure and (steady) long-exposure images. The authors successfully train a network to fully develop extreme low-light RAW images. The network learns the whole development pipeline (including denoising, demosaicking, tone mapping, etc.) and yields a significant improvement of the appearance of the output images. Other such real world datasets exists such as the SIDD dataset [1] and the DND dataset [45] for image denoising. The main limitation of such approaches is that for each new aquisition device, a new dataset needs to be collected, which is a tedious and time-consumming process.

In order to circumvent these heavy acquisition campaigns, another approach is to build realistic degradation models. At sensor level, noise can be modelled

by the sum of a signal dependant component (the shot noise) and an electronical noise (the readout noise). To model them, the common practice is to use a Poisson-Gaussian approximation parametrized by the gain (the ISO). However, when the ISO increases too much or when in extreme low-lights conditions, this model becomes inaccurate and poorly correlates with observations. The authors of [60, 61] propose a noise model for such extreme low-light conditions. Another case were modeling is complicated is when dealing directly with developped RGB images, in which noise is correlated between different color channels [41] and modified by non-linear tone mapping operators. In such cases, an interesting approach is to artificially unprocess the images and add noise in an approximated RAW domain [7]. Denoising networks trained with such sophisticated degradation models often perform on par with the same networks trained with real world data [58]. A global takeaway from the litterature is that training with a variety of degradations at different levels of intensity allows for better generalization, whether for super-resolution [64] or image denoising [7, 63].

2.3 Synthetic image models and deep learning

As explained above, the goal of this paper is to develop fully synthetic image datasets to train neural networks. In this paragraph, we first review some of the classical generative models for natural images and then consider works in which synthetic images are used to train neural networks.

Modelling natural images. Texture synthesis has always been a fruitful framework for the development of generative image models, for instance Markov random fields [15], wavelet models [26, 46], Gaussian models [21]. The statistical study of natural images ([9, 51]) has also motivated the development of statistically faithful image generation algorithms. Among the well-known property of natural images, let us mention scaling property and the non-Gaussianity of most observed statistics [40]. The first property can for instance be observed through the power law distribution of the power spectrum [49]. The second one through wavelets coefficients that are adequately modeled with generalized Gaussian distributions [46]. Dead leaves models ([4, 13, 24, 35, 38]), in particular, allows the reproduction of such behaviors with a very small number of parameters. Fractals models have also been used to model natural images, having a multi-scale and self-similar structure ([44, 48, 55]).

In the past decade, learning based approaches have led to great progresses, relying on statistics of CNN features [22], Generative adversarial networks (GANs) [23,30], Normalizing flows [33] or diffusion models ([27,52]). Such models requires huge databases to be trained and involve a very large number of parameters (for instance, models from [22] involves several thousands of parameters).

Training neural networks with synthetic data. In this paper, we show that image restoration networks can be fully trained from synthetic dead leaves images. To the best of our knowledge, this idea was first proposed in our preliminary work [2]. Other works have already tried to train network with synthetic images. This is especially appealing since one can easily generate as much data as needed, while having some control on the desired properties of the images.

For low-level vision tasks such as optical flow or depth prediction, 3D rendered scenes were shown to be sufficient to train neural networks [10, 20, 39, 54]. However, these synthetic sets require a careful design to be somewhat realistic. Going a step further, and extending [2], [37] proposes to train a disparity predictor from a 3D dead leaves model, achieving impressive results. Synthetic trainings have also been developed for more high-level computer vision tasks such as image classification. Fractal models are proposed in [31] as an efficient pre-training and a variety of simple random processes, including dead leaves, have been investigated in [5], showing very promising results when fine-tuned on different datasets.

3 Dead leaves images

The dead leaves model was originally introduced by the mathematical morphology school, with the aim of modeling porous media [38]. Despite being a particularly simple model, it was later shown to account for many statistics of natural images [4, 35]. Its structure is inherited from the sequential superimposition of random shapes, thereby mimicking a simplified image formation process, in which closer objects hide further ones. In this section, we first recall the mathematical definition of the model [6, 25], before detailing the algorithm and parameters that we will use to generate a synthetic image training dataset.

3.1 The continuous dead leaves model

The dead leaves model is defined from a set of random positions, times and shapes $\{(x_i, t_i, X_i)_{i\in\mathbb{N}}$, with $\mathcal{P} = \sum \delta_{x_i,t_i}$ a stationary Poisson process on $\mathbb{R}^2 \times (-\infty, 0]$ and the X_i are random sets of $\mathbb{R}^2$ that are independent of $\mathcal{P}$. The sets $x_i + X_i$ are called *leaves* and for each i, the *visible part* of the leaf is defined as

$$V_i = (x_i + X_i) \setminus \bigcup_{t_j \in (t_i, 0)} (x_j + X_j),$$

where by definition $A \setminus B = A \cap B^c$, with B^c the complementary set of B. That is, the visible part of leaf (x_i, t_i, X_i) is obtained by removing from this leaf all leaves $x_j + X_j$ that are indexed by a time t_j greater than t_i (that falls after it). The dead leaves model is then a tessellation of the plane, defined as the collection of all visible parts. A random image can be obtained from this tessellation by assigning a random color to each visible part. In the following of this paper, the term *dead leaves* model will refer to this random image. Examples of dead leaves models can be seen in figure 3. The example in Figure 3a is a simple example where the leaves are disks with constant radius. Such a model (only depending on one parameter, the disk size) already mimics two important property of natural images, namely the presence of edges and homogeneous area. Nevertheless, it lacks details.

In order to get more faithful synthetic images, it has been shown that one could use power functions for the distribution of objects sizes [4, 35]. This way,

the resulting images inherits scaling property and have been shown to reproduce many statistical properties of natural images. Such models are obtained by considering random leaves $R.X$, where X is a given shape and R is a real random variable with density $f(r) = C.r^{-\alpha}$, with C a normalizing constant. The case $\alpha = 3$ corresponds to a scale invariant model [35]. In order for such models to be well defined, values of R have to be restricted to values in (r_{min}, r_{max}) [25], resulting in a model with 3 parameters: r_{min}, r_{max} and α.

This model is especially appealing for natural images, because it incorporates two of their most fundamental property, non Gaussianity (as a result of edges) and scaling properties [40], in a very simple setting. Because this model contains details and edges at all scales, potentially of arbitrary contrast, it has been proposed as a tool for the evaluation of the ability of imaging devices to respect textures [12, 13] and was recently retained as a standard for quality evaluation [29].

3.2 The generation algorithm

Algorithm 1: Dead leaves image generation algorithm

Parameters: $(r_{min}, r_{max}, \alpha, w)$, color_image
Output : X
mask = ones(w, w);
X = zeros$(w, w, 3)$;
while $||mask|| > 0$ **do**
  tmp = $r_{max}^{1-\alpha} + (r_{min}^{1-\alpha} - r_{max}^{1-\alpha})\times$random();
  r = tmp$^{-\frac{1}{\alpha-1}}$;
  x,y = randint(0,w), randint(0,w);
  color = color_image(randint(0,w), randint(0,w));
  new_disk = disk(r, color);
  X = update_image(X, x, y, new_disk);
  mask = update_mask(mask, r, x, y);
end
X = downscale(X,5).

We now detail how to generate digital samples of the dead leaves model, following the procedure summarized in Algorithm 1. At each step, a random discrete disk of radius r and center (x, y) is generated as the set of discrete positions satisfying the corresponding disk equation. Centers are uniformly distributed in the image domain and radiuses are distributed according to a power law density with exponent α, as discussed in the previous paragraph. Radiuses are limited between r_{min} and r_{max}. To generate the image, we rely on a *perfect simulation* technique [32] and sequentially put the disks *below* the previously drawn disks, until the image domain has been fully covered. That is, at each step, pixels which have not been colored yet are given the color of the disk added at

this step. The choice of the disk color will be shortly discussed. The used definition of discrete disk is crude and in particular does not include any anti-aliasing scheme. Therefore we first generate a large image that is then downsampled by a factor 5 after convolution with a Gaussian filter with $\sigma = 5/3$ (roughly ensuring Shannon conditions). This step is a critical component of our algorithm. It allows for sub-pixel sized objects and for more natural boundaries. In Figure 2a, we display a full size (2000,2000) dead leaves image before downsampling. A (20,20) crop on that same image (see Fig. 2b) exhibit very sharp boundaries and piecewise constant zones. A (20,20) crop on the downsampled image has a more realistic aspect (see Fig.2b). The whole procedure can be seen as a very simple simulation of the camera acquisition of a dead leaves model with tiny objects.

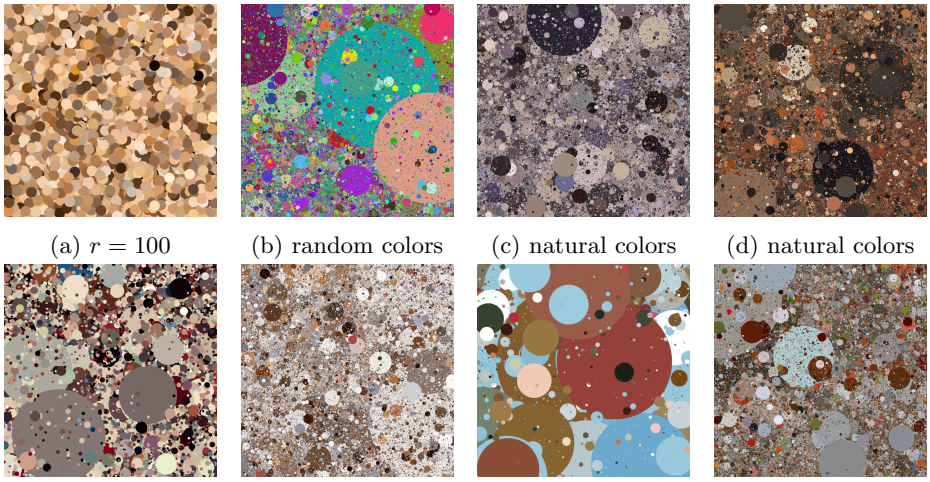
(a) (2000,2000) image (b) (20,20) crop (c) (20,20) crop after down sampling

Fig. 2: Illustration of the down sampling step (ds)

Color sampling. Our aim is to produce synthetic image databases accounting for the statistics of natural images. In particular, the marginal of color distribution should be as faithful as possible. In order to do so, we sample the colors of disks from natural image databases. As we shall see, this yields better restoration performance than sampling colors uniformly in the RGB cube. We can see in Figure 3 that Fig. 3c,3d have more realistic colors than Fig. 3b (uniform distribution prior on colors). In fact, for each generated dead leave image, we sample the colors for *a single* image. This yields more coherence in the color of the generated images and also improves performance. This last fact is indeed an interesting observation, since it suggests that neural networks benefit from color combinations that are likely to be encountered in natural images.

Size parameters. Since the shape of the leaves is fixed in our model (these are disks) the geometry of the generated images is solely controlled by the parameters of the size distribution, r_{min}, r_{max}, α. Images generated with different parameters can be seen in Figure 3. As we can see, at fixed $\alpha = 3$ and $r_{max} = 2000$, the visual appearance strongly depends on the value of r_{min}. A large value of r_{min} yields structured images, with visible edges and homogeneous zones, whereas smaller r_{min} yields texture-like, cluttered images (see Fig. 3e,3f). Similar observations

can be made when varying the scale parameter α (see 3g,3h). In our experiment, we chose the value $\alpha = 3.0$, which corresponds to scale-invariance. We also chose to fix r_{max} and to vary only r_{min}, which enables to set the structure/texture balance of the image.



(a) $r = 100$ (b) random colors (c) natural colors (d) natural colors

(e) $r_{min} = 20$ (f) $r_{min} = 1$ (g) $\alpha = 2.0$ (h) $\alpha = 3.0$

Fig. 3: Dead leaves images generated with different parameters.

4 Noise models and synthetic image databases

In this section, we consider three different noise models corresponding to increasingly difficult settings : a naive RGB model, a model for RAW images captured by smartphones and a model for extreme low light RAW images. These situations correspond to the cases that we will consider in the experimental section. We also present the degradation model for the super resolution problem. For each model, we also detail the generation of the corresponding synthetic dead leaves models that we will use to train different neural networks.

4.1 Naive additive model for RGB images

Noise model. The simplest and most used noise model in the literature it the additive white Gaussian noise (AWGN) model, where the noisy image is the sum of the ground truth image x and a white Gaussian noise of variance σ^2:

$$y = x + n, n \sim \mathcal{N}(0, \sigma^2).$$

Although this model is limited and mostly wrong in real world situations, it is a good starting point to test the possibility of training a network with synthetic images.

Image dataset. The corresponding image dataset is made of clean and noisy dead leaves images. In order to account for both homogeneous areas and micro-textures, we build a dataset made of images generated with either $r_{min} = 1$ or $r_{min} = 16$, in both cases combined with parameters $\alpha = 3.0$ and $r_{max} = 2000$. Micro-textures being harder to restore than homogeneous areas, we chose to have a 2 to 1 ratio between the two possible r_{min} values. The color distribution of the disks is obtained by sampling a database of natural images, as we will see in the experimental section. As shown previously, this leads to a more coherent color distribution than randomly sampling the RGB cube. Finally, we apply a Gaussian blur to a 10th of the dataset, with a standard deviation uniformly sampled between 1 and 3. Indeed, most natural images tend to contain blurry zones due to the limited depth-of-field of cameras. By adding this very simple blur model to some of the images of the dataset, we observed that blurry areas in natural images can be better restored.

4.2 Single image super resolution for RGB images

For that task, the goal is to recover a high resolution image from low resolution one. Usually, the deterioration of a high resolution image x is modelled by a simple downsampling with a bicubic interpolation of a given factor. Our degraded observations y are the results of that process :

$$y = B(x, \lambda),$$

where B is the bicubic downsampling operator and $\lambda \in [2, 4]$ the downsampling factor. Our synthetic dataset for this task is exactly the same as our AWGN denoising dataset, with the same amount of micro-textures and homogeneous areas.

4.3 Noise removal on RAW smartphone images

Noise model. As we mentioned in Section 2.2, image denoisers trained with AWGN fail to denoise real noisy images. Indeed, in the RGB domain, noise is signal dependant and spatially correlated. When considering RAW images, the noise is (mostly) i.i.d. but signal dependent, making the AWGN model of limited use. For such images, the noise can be modeled with a signal dependant component (the shot noise), which accounts for the quantum behaviour of photons, and a signal invariant component (the read noise), which accounts for all the electronic noises that mostly occur when measuring the electrons produced by the sensor. Having a theoretically clean RAW image X, a real observation Y can be written as :

$$Y = X + N_s(X) + N_r,$$

where N_s stands for the shot noise, and N_r for the read noise. To model the photonic noise, a common practice is to use a Poisson law. While most papers use the Gaussian-Poisson approximation, [60] showed that this approximation does not stand in low-light conditions.

At sensor level, let's denote by I_t the matrix (image) of expected number of photons. Due to the Poisson behaviour of photons the number of photons actually measured is $I \sim \mathcal{P}(I_t)$. Note that, contrary to I_t, I is an integer which is crucial in the case of very low-light conditions. [1] In order to retrieve a RAW image from this photon count, I is multiplied by a digital gain K that depends on the ISO at shooting time. The RAW noisy image is therefore $Y = KI + N_r$. Following these equations, we can generate a noisy image Y from a noiseless image X, by retrieving the initial photon count $I = X/K$. A noisy simulation of the shot noise can be obtained with the following equation :

$$X + N_s(X) \sim K \times \mathcal{P}(X/K),$$

where K depends on the ISO.

The read noise N_r accounts for all electronic noise sources. To generate it, we chose the Gaussian approximation, which is the most common and reliable model :

$$N_r \sim \mathcal{N}(0, \sigma^2),$$

where the standard deviation σ also depends on the ISO.

In order to generate our synthetic image dataset, the last step is to estimate the parameters of the noise model, which are device dependent. In the experimental section, we will consider the problem of denoising images from the SIDD dataset [1]. This database is composed of RAW images from 5 different phones at different ISOs, in controlled lighting conditions. In order to estimate the parameters for each device, we make use of the ground truth images from this base. These images are obtained by averaging 400 shots of the same scene in identical conditions. The authors provide noise parameters along each pictures, but for a Gaussian-Poisson approximation of the shot noise. Unfortunately, the parameters provided were also incorrect : for example, the read noise was supposed to be null for some cameras. Given the dataset, we could properly estimate those parameters in a procedure described in appendix A.

Synthetic database. In order to generate a synthetic image database with the same characteristics as RAW images, we combine the previous noise model with clean RAW dead leaves images. In order to create these non-demosaicked images, we first sample colors from a Bayer frame of a ground truth image. To do so, we pack RAW images in a 4 channel R-G1-G2-B tensor as explained in Figure 4a.

Colors are sampled at random (x, y) positions on this tensor. We create a dead leaves image of size $(4H, 4W, 4)$ with the same procedure than for RGB images. However, it is important at this stage that the green components of a single color should be equal, otherwise there would be checkerboard artifacts

[1] Actually one has no access to the measure I but to the proportional measure which the number of electrons produced for each digital value.

(a) Bayer to Tensor operation (b) Tensor to bayer filtering

Fig. 4: Bayer frame manipulations

on the final developed image. Therefore, we replace both green components by their average. After blur and downscaling, we are left with a $(H, W, 4)$ dead leaves tensor. In order to go back to the RAW domain which has only one channel, we filter the created tensor following the color-filter array (CFA) of the camera of the source image, as shown in Figure 4b. That is, at each position (x, y) in the array, we only keep the value of the color at the same position in the Bayer frame. The size of the final RAW dead leaves image is now $(H, W, 1)$.

To account for both textures and homogeneous zones as well as blurry areas, we generated half of the images with $r_{min} = 4$, and the other half with $r_{min} = 100$. We applied a Gaussian blur to one third of the images with a kernel standard deviation uniformly sampled in $[1, 3]$.

4.4 Low-light image enhancement

Noise model. In extreme low-light condition, the noise model becomes more complex. In [60], the authors provide an extensive study of noise in such conditions for the Sony α7s2 camera, which was used to collect the See-in-the-Dark dataset [14]. According to them, the shot noise can still be modelled by a Poisson noise, but the read noise can not be modelled as a zero-mean Gaussian noise. They model the read noise as a Tukey-Lambda distribution, and estimate its parameters in a complex protocol. A key ingredient of their success, was to add an estimated offset μ to their previous zero-mean noise. Since the authors do not provide these parameters, we chose for simplicity an uncentered Gaussian approximation of the read noise :

$$N_r \sim \mathcal{N}(\mu, \sigma^2),$$

where μ and σ both depends on the ISO and were estimated with a simple protocol that we describe in the appendix A. In low-light conditions, random banding patterns are visible due to a readout electronic noise. It is modelled by a random offset added to each row of the clean RAW image. This banding noise N_b follows a centered Gaussian distribution $\mathcal{N}(0, \sigma_r^2)$, where σ_r also depends on the ISO. Finally, the image is quantized inducing quantization noise N_q. Therefore, our noisy image follows this equation :

$$Y = X + N_s(X) + N_r + N_b + N_q.$$

Image generation specifics. The generation algorithm is almost the same as the one used for smartphone camera denoising. We sample the colors in the ground-truth long exposure RAW images to create our RAW dead leaves image. This image is then developed with dcraw to obtain a clean RGB target. To obtain our synthetic short exposure image, we start by subtracting the black level to the RAW dead leaves image. Then we divide it by a factor 100,250 or 300. These are the usual ratios between short and long exposure times in the dataset. At this stage, noise is added following our model. The image is then multiplied by the exposure ratio that we first chose to increase the image dynamic.

Concerning the generation parameters, we use the same parameters we chose for real-world image denoising.

5 Results

5.1 Additive white gaussian noise removal for RGB images

Network and Training. We start by the simplified task of denoising RGB images corrupted with AWGN. In order to assess the training capacity of the proposed synthetic dataset, we consider the widely used network FFDNet, a state-of-the-art image denoising CNN which was introduced by Zhang et al. [63] and thoroughly examined in [53]. Its main specificity relies in the first layer of the network : to increase the receptive field and to handle a wide range of noise levels, the image is divided in four sub-images which are concatenated to a noise map indicating the local noise standard deviation. This tensor is then passed through a more classic network of batch normalized convolutional layers, with an architecture similar to that of DNCNN's [62]. The network then outputs the four denoised sub-images, which are reassembled to create the final denoised image.

To compare different trainings fairly, we always use the same optimization algorithm. It consists of 80 epochs with the Adam optimizer and the L2 loss, starting with a 10^{-3} learning rate. There is a decay of factor 10 at epoch 50, and another decay of factor 100 at epoch 60. For each training, we used 350k $(50, 50, 3)$ patches.

Comparison. After training FFDNet on our synthetic dataset, we compare its results to those of a training on natural images and of a training on a mixed dataset. The latter contains $\frac{1}{3}$ dead leaves images, and $\frac{2}{3}$ natural images. To show that scaling properties are needed to model natural images, we also trained FFDNet on dead leaves images generated from disks with a fixed radius of 100. In addition, we also consider two alternative training schemes from datasets of synthetic images : white noise images and Gaussian random fields [21] (see Figure 5). Numerical evaluation is performed on 2 test sets of natural images (CBSD68, Kodak24) and one set of 24 dead leaves images, generated using the colors of the dataset Kodak24.

For each test, we compute the average PSNR, SSIM [59] and PieAPP metric [47], a recent perceptual metric based on human annotation.

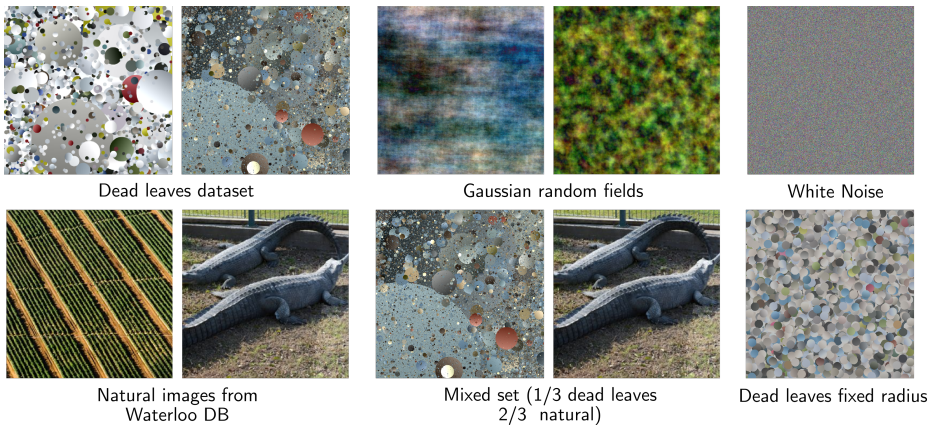


Fig. 5: Example images of our different training sets for FFDNet

Table 1: Numerical comparisons of the different trainings of FFDNet. We evaluated the results on two benchmark datasets for image denoising (CBSD68 and Kodak24), and our dead leaves testset, at two noise levels. Each cell contains the triplet PSNR/SSIM/PieAPP. The best results are in blue, the second best in red.

σ	Dataset	CBSD68	Kodak24	Dead leaves testset
25	White Noise	19.52/0.416/2.386	19.68/0.365/2.502	20.36/0.607/2.043
	Gaussian field	29.63/0.845/1.402	30.24/0.835/1.471	26.23/0.826/1.254
	DL $r = 100$	29.56/0.820/1.218	30.49/0.819/1.024	26.13/0.799/1.263
	Dead leaves	30.58/0.867/0.711	31.27/0.859/0.739	27.46/0.865/0.573
	Mix	31.07/0.881/0.639	31.98/0.876/0.603	27.33/0.860/0.567
	Natural Images	31.09/0.882/0.629	32.00/0.878/0.599	27.05/0.851/0.576
50	White Noise	15.58/0.247/4.682	15.71/0.209/4.785	16.24/0.387/2.932
	Gaussian field	26.68/0.738/2.203	27.41/0.737/2.353	23.31/0.694/2.158
	DL $r = 100$	26.85/0.720/1.563	27.91/0.739/1.314	23.24/0.654/2.005
	Dead leaves	27.40/0.762/1.088	28.21/0.765/1.154	24.21/0.737/1.020
	Mix	27.86/0.782/0.997	28.86/0.789/0.985	24.12/0.732/1.015
	Natural Images	27.87/0.786/0.991	28.89/0.792/0.978	23.90/0.722/1.053

On both natural image testsets and on the dead leaves testset, we observe that the model trained on dead leaves outperforms by a large margin all other models trained on alternative synthetic image datasets (0.9db for the Gaussian model and, without surprise, 11 dB for the white noise model), see Table 1. Visually, the Gaussian field model leads to denoised images still containing noise and grid-like artifacts, which severely impact the PieAPP metric. Observe that for both image models of white noise and Gaussian noise, the optimal solution is known and given by the Wiener filter (multiplication by a constant in the first case and linear filtering in the second). It is worth mentioning that the network did not learn to apply this theoretical optimal solution to natural images in either cases. Confirming our intuition that an image model with scaling properties is needed, the dead leaves model with a fixed radius tends to strongly over-smooth the image, thus losing all texture information. This amounts to a loss of 0.65 dB on natural image testsets, and 1.2 dB on dead leaves images.

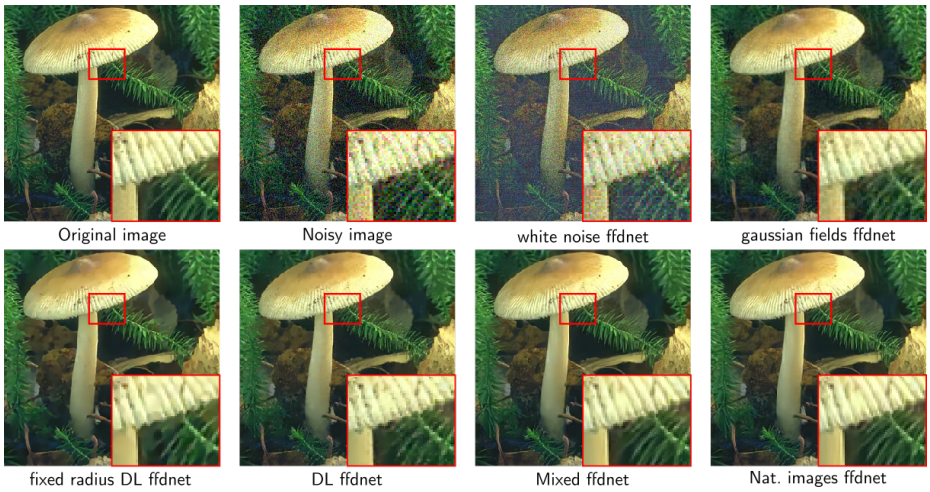


Fig. 6: Denoising comparison with different FFDNet trainings.

More surprisingly, the model trained exclusively on dead leaves images performs only 0.6dB lower than the model classically trained on natural images. Visually, the results are still almost as good, despite some limitations. In particular, the synthetically trained model has some difficulties with thin and low contrast lines, and occasionally creates dot artifacts. In other situations, the synthetic training improves the results, as can be seen in Figure 6, where one can notice a better preservation of fine details (other than thin lines) on the texture on top of the mushroom.

Another interesting result is the fact that training on a mix of dead leaves and natural images improves the denoising of the first category without imparing the denoising of the second.Indeed, on testsets of natural images, the difference

in PSNR being less than 0.02dB. Visually, the results are almost identical, with a slight advantage for the mixed trained model on texture areas. On the dead leaves test set, the mixed trained model clearly outperforms the model trained on natural images, by 0.25dB. This result suggests that jointly optimizing the response to this kind of mixed datasets has the ability to increase some aspects on which imaging devices are evaluated. Indeed the scaling dead leaves model is classically used to evaluate the ability of imaging devices to preserve texture areas [12,13] and the corresponding scale-invariant test chart has recently become an ISO standard [29].

Ablation study. To confirm the choices made to build the synthetic dataset, we compare different trainings performed with different parameters or design choices, both visually and numerically.

We first illustrate the impact of r_{min} on the denoising results. As shown in Figure 7, the smaller r_{min}, the better micro-textures are restored. Conversely, they are smoothed when r_{min} gets larger. On the other hand, homogeneous regions contain artifacts when r_{min} is too small, and are well restored when r_{min} is larger. This behaviour is expected since a large r_{min} leads to dead leaves images with homogeneous regions, and a small r_{min} to more textured regions. Referring to Table 2, the optimal r_{min} seems to be around 4. However, by mixing images generated with $r_{min} = 1$ and $r_{min} = 16$, we get a noticeable improvement in PSNR (0.17dB) and in image quality, as can be seen in Figure 7.

Table 2: Impact of the parameters and ablation study. In the first 3 columns(DL1 to DL16), we fix the parameters to $r_{max} = 2000$, $\alpha = 3.0$, with natural colors and the downscaling step. From column 4 to 7, we keep the same parameters as in the final dataset, but we remove some important features of the generation. The last column corresponds to the final result.

σ	DL-1	DL-4	DL-16	Rand. col	No sub	No blur	Final
25	31.03	31.09	30.98	29.99	30.79	31.25	31.27
50	27.98	28.04	28.05	27.16	27.74	28.20	28.21

Other important features of the synthetic images generation are : the color distribution, the downscaling step, and the blur. As we can see in Figure 7, when we sample the disks colors uniformly in the RGB cube, the denoised images show many color artifacts. The additive Gaussian noise creates unnatural colors that the network doesn't identify as such, since it has not been trained on images with natural colors. This leads to a performance gap of more than 1 dB in PSNR. The downscaling step is also critical, as it allows sub-pixel sized objects and more natural boundaries. In Figure 7, we can see that the network trained on the dead leaves dataset without subsampling tends to over-smooth texture areas, and to produce stair-casing artifacts. We can also identify some disk-like objects with hard boundaries in the images, creating an unnatural aspect. In terms of PSNR, this amounts to a loss of 0.5 dB compared to the training on our final dead leaves dataset. For the final synthetic dataset, we decided to blur 10 % of images. As

we can see in Table 2, removing this step has almost no impact on the PSNR. Nonetheless, we observe in Figure 7 that removing this step makes blurry zones look sharper than they really are. Overall, this shows that each step of the image synthesis algorithm is important to get results that are numerically and visually satisfying.

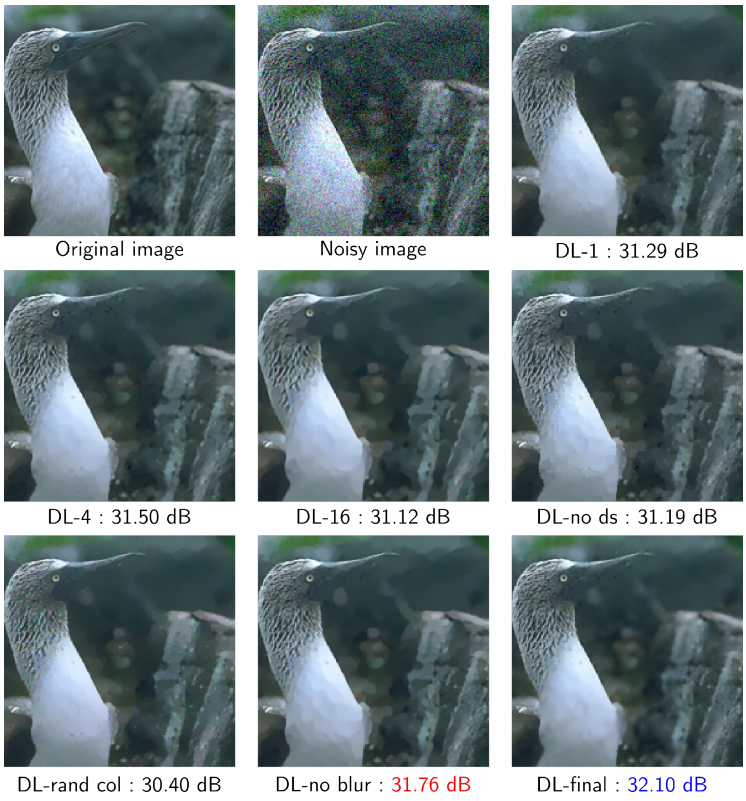


Fig. 7: Visual illustration of the ablation study. From left to right : First line - clean image, noisy image, $r_{min} = 1$ / Second line - $r_{min} = 4$, $r_{min} = 16$, no subsampling / Third line - uniform color distribution, no blur, final result.

5.2 Single image super resolution for RGB images

Network and Training. Next, we turn to the task of super resolution. We chose to retrain the Residual Dense Network (RDN) [65], a classical super-resolution network. Its architecture is based on residual dense blocks, a combination of dense blocks introduced in [28] and residual connections. The model trained on dead leaves (DL-RDN) is trained with the same dataset used for AWGN removal. The model trained on natural images (called Nat-RDN) is trained on a portion

of the DIV2K dataset. The training is performed for 800 epochs with a batch-size of 16. We optimize the L_1 distance between the predicted image and the high resolution ground truth with the ADAM optimizer. We based our experiments on an un-official yet exact github implementation.[2]

The numerical evaluation shows a similar behaviour to the one observed for AWGN removal presented in Section 5.1. The loss in performance when using the dead leaves model is of 1.2dB and 0.6dB for a super-resolution of scale 2 and 3 respectively. The results on the Set5 and Set14 datasets, which are common benchmarks for super-resolution, are given in Table 3.

Table 3: Numerical evaluation of our Super-resolution results. We report the PSNR of RDN trained either on the dead leaves dataset or on the DIV2K dataset [3].

Dataset	Set 5		Set 14	
scale	×2	×3	×2	×3
Dead leaves	36.76	33.82	32.93	30.42
Natural Images	38.18	34.71	33.88	30.73

Visually, the super-resolution results are similar between the two trainings as we can see in Figure 8. Looking closer to the details of the restored images, we see that the output of DL-RDN tends to reproduce better the white dots in the butterfly wing and the texture of the bird's beak. Conversely, the thin lines in the yellow regions of the butterfly wing have a "dotted" aspect in the DL-RDN image, whereas they are properly restored in the Nat-RDN image. Indeed, the dead leaves model does not contain any straight and thin lines, making it harder for this model to retrieve them. Enriching the dead leaves model with elongated shapes or adding different local structures in the training are perspectives to solve this issue.

5.3 Noise removal on Smartphone RAW images

Network and Training. Next, we consider a real case scenario, by considering the denoising of smartphone RAW images from the SIDD dataset [1]. To denoise RAW images, we adapted the U-Net architecture to our problem. First, we modified the input layer, so that it transforms a Bayer frame in a RGGB tensor as shown in Figure 4a. This tensor is passed to the convolutional network. Rather than using transposed convolutions in the decoder part, we preferred a bilinear upsampling followed by a convolutional layer. This is done to avoid checkerboard artifacts which were thoroughly investigated in [42]. We also chose a residual approach, meaning that the network outputs a prediction of the noise, rather than the clean image. This technique helps to improve the convergence speed, and was also used for FFDNet.

To train our network, we generated a dataset of 16000 RAW dead leaves patches of size (256,256). We also trained this network with the same number of

[2] https://github.com/yjn870/RDN-pytorch

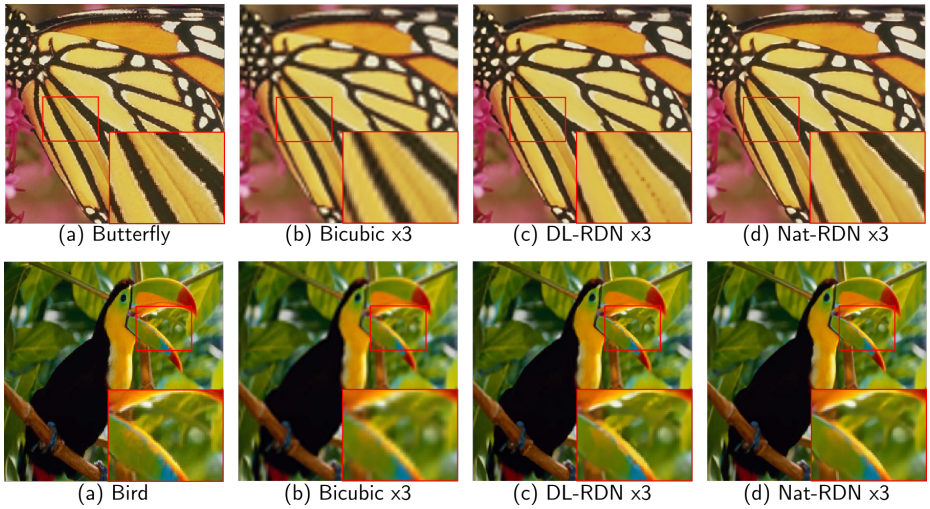


Fig. 8: Visual comparison between the two differently trained RDN approaches, together with a simple super resolution baseline, for a zoom factor of 3. From left to right : High resolution image, Bicubic interpolation, RDN trained on dead leaves images, and RDN trained on natural images.

patches taken from natural images of a 80% portion of the SIDD-Small dataset in order to compare natural and synthetic data. The other 20 % were used for our test set. To assess the truthfulness of our noise model, presented in Section 4.3, we compare two trainings of the model on natural images : one using synthetic noise and the other with real noisy data. We optimize the L_1 loss during 1000 epochs with the Adam optimizer. The learning rate is 10^{-4} for the first 500 epochs, 10^{-5} for the following 300 epochs, and 10^{-6} for the last 200.

Results. To evaluate our models, we tested them on the remaining 20% of the SIDD small dataset. Since the network produces a RAW output, we develop it with a simple pipeline proposed by the authors of the SIDD dataset. The proposed pipeline is made of a sequence of operations : white balance using metadata of the sensor, edge-aware demosaicing, a color space transform from RAW colors to sRGB, a gamma correction and a simple tone-mapping function. Given this pipeline, we can measure PSNR in both the RAW and RGB domains.

The numerical results presented in Table 4 show that training with dead leaves images and a synthetic noise model (DL-Unet) is as good as training with natural images and a synthetic noise model (RS-Unet). This further validates the hypothesis that synthetic data can be used to successfully train a complex image restoration network. More precisely: when testing on RAW images, the loss implied by using dead leaves instead of real images is of only 0.25 dB. When evaluating the result on the developed RGB images, the dead leaves training even outperforms the training on natural images, all other things being equal.

Further, we compare the performance of DL-Unet, the network trained on fully synthetic data, to the performance of the network trained with real noisy images (RR-Unet). The performance loss in this case is of 0.8 dB in the RAW domain, and 0.27 dB in the RGB domain. This shows that the full synthetic training is successful even though slightly less efficient than when using the ground truth. Actually, this loss in performance is of the same order as the loss when comparing RR-Unet and RS-Unet, showing that the loss is mostly due to the noise model and not to the image model.

Table 4: Numerical evaluation of the denoising of smartphone RAW images. We report the PSNR of our adapted Unet trained either on the dead leaves dataset or on the SIDD-small dataset with real or synthetic noise.

Dataset	SIDD-test RAW	SIDD-test RGB
DL images + synthetic noise (DL-Unet)	49.60 dB	38.05 dB
Nat. images + synthetic noise (RS-Unet)	49.85 dB	37.95 dB
Nat. images + real noise (RR-Unet)	50.40 dB	38.32 dB

Now, we argue further that *the synthetic training can in fact outperform the natural training*. First, this behavior is observed on the RGB results just presented. Moreover, if we look at Figure 9, we notice that noise is still present in the "ground truth" image for high ISOs. In both examples, the details show that averaging 400 images is not sufficient to fully suppress noise. The images denoised with DL-Unet appear cleaner in flat areas than the other denoised images which still contain some noise. Indeed, the simulated dead leaves contain noiseless flat areas. Moreover, the highlighted detail of the first image also shows some structures that appear sharper in the images denoised with DL-Unet. Even if the quality of these images seems superior, this is not fully reflected by the quantitative comparison, mostly because our synthetic noise model is not perfect and also possibly because of the limitations of the PSNR to quantify image quality.

5.4 Low-light image enhancement

Network and Training. Eventually, we consider the task of denoising images taken in extreme low light. We consider the See-in-the-Dark dataset and chose to train an adapted version of a U-Net, similar to the one we used for the smartphone database. The RAW input is packed in a R-G1-G2-B tensor, and passed through the UNet, which uses bilinear upsampling and convolutions rather than transposed convolutions in the decoder part of the network. To train our network DL-UNet, we generated 10000 RAW clean dead leaves images of size (256,256), and the corresponding noisy images using the synthetic noise model presented in Section4.4. The training algorithm is the exact same that was used for smartphone images denoising. To sample the noise parameters, we use the frequency of each ISO in the Sony database, given that each set of parameter corresponds

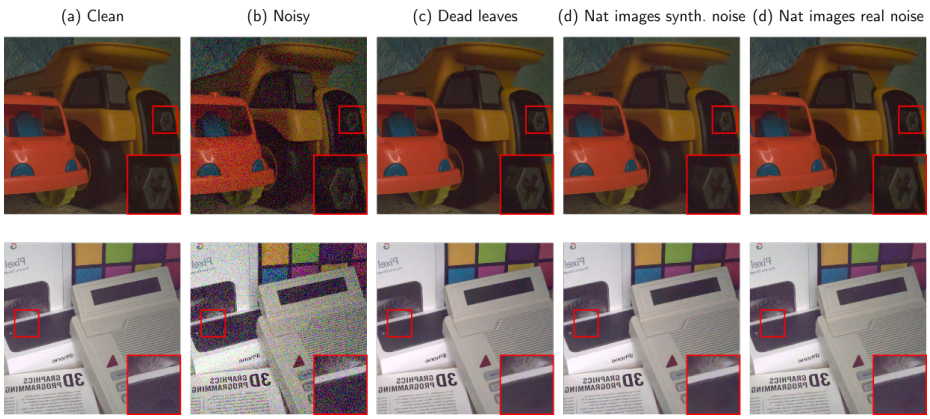


Fig. 9: Real denoising results.

to a unique ISO. We expect that by sampling the noise parameters this way, the network will be able to restore properly images at varying ISOs, and to generalize well to the test set. For simplicity the exposure time ratios are uniformly sampled between $(100, 250, 300)$. To assess the validity of our noise model, we train the network also on clean natural images to which we add our synthetic noise model(RS-UNet). We then compared these models to the original model trained on real noisy and clean image pairs (RR-UNet).

Results. We tested our trained models on the See-in-the-Dark test set, made of 94 RAW images at different ISOs and different time exposure ratios, with indoor and outdoor scenes. We report in Table 5 the quantitative results of our tests.

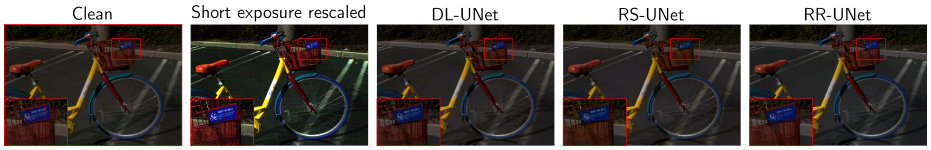
Table 5: Numerical evaluation of our low-light enhancement networks. We report the PSNR of our three different models at different time exposure ratios. In last three lines, we also report a corrected PSNR where we prescribe the real average of each channel to the output image.

ratio	100	250	300	Global
DL-UNet	28.90	26.86	26.62	27.41
RS-UNet	29.74	27.28	26.80	27.87
RR-Unet	29.81	28.55	28.21	28.83
DL-UNet (corrected mean)*	30.08*	28.28*	27.15*	28.42*
RS-UNet (corrected mean)*	30.18*	27.97*	27.32*	28.42*
RR-Unet (corrected mean)*	29.81*	28.55*	28.21*	28.83*

In terms of PSNR, we notice that the DL-UNet has a 1.4 dB difference with the RR-UNet which is a significant margin. However we notice that even RS-UNet has almost a 1dB gap with RR-UNet. This shows that our main limitation does not reside in the image generation algorithm but in the degradation model. For high ISOs and extreme low light indoor photographs, the noise model is indeed very complex and difficult to model. We also observed that the models

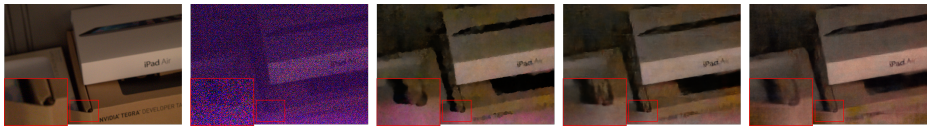
trained with a synthetic degradation model sometimes fail to produce a well white-balanced image, resulting in low PSNR scores. That is why we also report a corrected PSNR, where we prescribe the mean of each color channel of the output image to the mean of its clean long exposure counterpart. After applying this post-processing, we see that DL-UNet performs on par with RS-UNet and only 0.4 dB behind RR-UNet, which is a small margin. We believe that this metric allows for a better assessment of the denoising quality of our model.

Visually the results of the DL-UNet are close and sometimes better than RS-UNet's and RR-UNet's. Looking at Figure 10a, we see that the details of the bike's basket are sharper in the DL-UNet image than in the other restorations. In Figure 10b, we see that DL-UNet removes banding noise more effectively than RS-UNet in homogeneous areas, while preserving sharp details in the vegetation area. In more extreme low-light conditions (see Figure 10c), we see that even if the result is close to the one of RR-UNet, DL-UNet tends to produce more color-artefacts and does not preserve straight lines as RR-UNet or RS-UNet do. In such conditions, our degradation model fails to mimic the sensor's behaviour. Indeed, we notice on some images that the lense produces ring like artifacts and that the bottom of the sensor is biased. Since we do not model these phenomena, our model fail to properly restore the image while creating artifacts.



(a) Comparison of our models on an outdoor image at ISO 250 and exposure ratio 100

(b) Comparison of our models on an outdoor image at ISO 250 and exposure ratio 300



(c) Comparison of our models on an indoor image at ISO 12800 and exposure ratio 300

Fig. 10: See-in-the-Dark test examples

References

1. Abdelhamed, A., Lin, S., Brown, M.S.: A high-quality denoising dataset for smartphone cameras. In: Proceedings of the IEEE Conference on Computer Vision and

Pattern Recognition. pp. 1692–1700 (2018)
2. Achddou, R., Gousseau, Y., Ladjal, S.: Synthetic images as a regularity prior for image restoration neural networks. In: Scale Space and Variational Methods in Computer Vision. Springer International Publishing (2021)
3. Agustsson, E., Timofte, R.: Ntire 2017 challenge on single image super-resolution: Dataset and study. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops (July 2017)
4. Alvarez, L., Gousseau, Y., Morel, J.M.: The size of objects in natural and artificial images. In: Advances in Imaging and Electron Physics, vol. 111, pp. 167–242. Elsevier (1999)
5. Baradad, M., Wulff, J., Wang, T., Isola, P., Torralba, A.: Learning to see by looking at noise. Advances in Neural Information Processing Systems **34** (2021)
6. Bordenave, C., Gousseau, Y., Roueff, F.: The dead leaves model: a general tessellation modeling occlusion. Advances in applied probability **38**(1), 31–46 (2006)
7. Brooks, T., Mildenhall, B., Xue, T., Chen, J., Sharlet, D., Barron, J.T.: Unprocessing images for learned raw denoising. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11036–11045 (2019)
8. Buades, A., Coll, B., Morel, J.M.: A review of image denoising algorithms, with a new one. Multiscale Modeling & Simulation **4**(2), 490–530 (2005)
9. Burton, G.J., Moorhead, I.R.: Color and spatial structure in natural scenes. Applied optics **26**(1), 157–170 (1987)
10. Butler, D.J., Wulff, J., Stanley, G.B., Black, M.J.: A naturalistic open source movie for optical flow evaluation. In: European conference on computer vision. pp. 611–625. Springer (2012)
11. Candes, E.J., Tao, T.: Decoding by linear programming. IEEE Transactions on Information Theory **51**(12), 4203–4215 (2005). https://doi.org/10.1109/TIT.2005.858979
12. Cao, F., Guichard, F., Hornung, H.: Measuring texture sharpness of a digital camera. In: Digital Photography V. vol. 7250, p. 72500H. International Society for Optics and Photonics (2009)
13. Cao, F., Guichard, F., Hornung, H.: Dead leaves model for measuring texture quality on a digital camera. In: Digital Photography VI. vol. 7537, p. 75370E. International Society for Optics and Photonics (2010)
14. Chen, C., Chen, Q., Xu, J., Koltun, V.: Learning to see in the dark. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3291–3300 (2018)
15. Cross, G.R., Jain, A.K.: Markov random field texture models. IEEE Transactions on Pattern Analysis and Machine Intelligence (1), 25–39 (1983)
16. Cruz, C., Foi, A., Katkovnik, V., Egiazarian, K.: Nonlocality-reinforced convolutional neural networks for image denoising. IEEE Signal Processing Letters **25**(8), 1216–1220 (2018)
17. Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.: Image restoration by sparse 3D transform-domain collaborative filtering. In: Astola, J.T., Egiazarian, K.O., Dougherty, E.R. (eds.) Image Processing: Algorithms and Systems VI. vol. 6812, pp. 62 – 73. International Society for Optics and Photonics, SPIE (2008). https://doi.org/10.1117/12.766355, https://doi.org/10.1117/12.766355
18. Donoho, D.L.: Compressed sensing. IEEE Transactions on Information Theory **52**(4), 1289–1306 (2006). https://doi.org/10.1109/TIT.2006.871582
19. Donoho, D.L., Johnstone, I.M., et al.: Minimax estimation via wavelet shrinkage. The annals of Statistics **26**(3), 879–921 (1998)

20. Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., Brox, T.: Flownet: Learning optical flow with convolutional networks. In: Proceedings of the IEEE international conference on computer vision. pp. 2758–2766 (2015)
21. Galerne, B., Gousseau, Y., Morel, J.M.: Micro-texture synthesis by phase randomization. Image Processing On Line **1**, 213–237 (2011)
22. Gatys, L.A., Ecker, A.S., Bethge, M.: Texture synthesis using convolutional neural networks. arXiv preprint arXiv:1505.07376 (2015)
23. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. Advances in neural information processing systems **27** (2014)
24. Gousseau, Y., Roueff, F.: The dead leaves model: general results and limits at small scales. arXiv preprint math/0312035 (2003)
25. Gousseau, Y., Roueff, F.: Modeling occlusion and scaling in natural images. Multiscale Modeling & Simulation **6**(1), 105–134 (2007)
26. Heeger, D.J., Bergen, J.R.: Pyramid-based texture analysis/synthesis. In: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques. pp. 229–238 (1995)
27. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Advances in Neural Information Processing Systems **33**, 6840–6851 (2020)
28. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4700–4708 (2017)
29. Photography − Digital cameras − Part 2: Texture analysis using stochastic pattern. Standard, International Organization for Standardization, Geneva, CH (2019)
30. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4401–4410 (2019)
31. Kataoka, H., Okayasu, K., Matsumoto, A., Yamagata, E., Yamada, R., Inoue, N., Nakamura, A., Satoh, Y.: Pre-training without natural images. In: Proceedings of the Asian Conference on Computer Vision (2020)
32. Kendall, W.S., Thönnes, E.: Perfect simulation in stochastic geometry. Pattern Recognition **32**(9), 1569–1586 (1999)
33. Kingma, D.P., Dhariwal, P.: Glow: Generative flow with invertible 1x1 convolutions. Advances in neural information processing systems **31** (2018)
34. Lebrun, M., Buades, A., Morel, J.M.: A nonlocal bayesian image denoising algorithm. SIAM Journal on Imaging Sciences **6**(3), 1665–1688 (2013)
35. Lee, A.B., Mumford, D., Huang, J.: Occlusion models for natural images: A statistical study of a scale-invariant dead leaves model. International Journal of Computer Vision **41**(1-2), 35–59 (2001)
36. Liu, D., Wen, B., Fan, Y., Loy, C.C., Huang, T.S.: Non-local recurrent network for image restoration. In: Advances in Neural Information Processing Systems. pp. 1673–1682 (2018)
37. Madhusudana, P.C., Lee, S.J., Sheikh, H.R.: Revisiting dead leaves model: Training with synthetic data. IEEE Signal Processing Letters (2021)
38. Matheron, G.: Random sets and integral geometry (1975)
39. Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T.: A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4040–4048 (2016)

40. Mumford, D., Gidas, B.: Stochastic models for generic images. Quarterly of applied mathematics **59**(1), 85–111 (2001)
41. Nam, S., Hwang, Y., Matsushita, Y., Kim, S.J.: A holistic approach to cross-channel image noise modeling and its application to image denoising. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1683–1691 (2016)
42. Odena, A., Dumoulin, V., Olah, C.: Deconvolution and checkerboard artifacts. Distill (2016). https://doi.org/10.23915/distill.00003, `http://distill.pub/2016/deconv-checkerboard`
43. Ono, S.: Primal-dual plug-and-play image restoration. IEEE Signal Processing Letters **24**(8), 1108–1112 (2017). https://doi.org/10.1109/LSP.2017.2710233
44. Pentland, A.P.: Fractal-based description of natural scenes. IEEE transactions on pattern analysis and machine intelligence (6), 661–674 (1984)
45. Plotz, T., Roth, S.: Benchmarking denoising algorithms with real photographs. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1586–1595 (2017)
46. Portilla, J., Simoncelli, E.P.: A parametric texture model based on joint statistics of complex wavelet coefficients. International journal of computer vision **40**(1), 49–70 (2000)
47. Prashnani, E., Cai, H., Mostofi, Y., Sen, P.: Pieapp: Perceptual image-error assessment through pairwise preference. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1808–1817 (2018)
48. Redies, C., Hasenstein, J., Denzler, J., et al.: Fractal-like image statistics in visual art: similarity to natural scenes. Spatial vision **21**(1-2), 137–148 (2008)
49. Ruderman, D., Bialek, W.: Statistics of natural images: Scaling in the woods. Advances in neural information processing systems **6** (1993)
50. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D: nonlinear phenomena **60**(1-4), 259–268 (1992)
51. Simoncelli, E.P.: 4.7 statistical modeling of photographic images. Handbook of Video and Image Processing **9** (2005)
52. Song, Y., Ermon, S.: Generative modeling by estimating gradients of the data distribution. Advances in Neural Information Processing Systems **32** (2019)
53. Tassano, M., Delon, J., Veit, T.: An analysis and implementation of the ffdnet image denoising method. Image Processing On Line **9**, 1–25 (2019)
54. Tremblay, J., Prakash, A., Acuna, D., Brophy, M., Jampani, V., Anil, C., To, T., Cameracci, E., Boochoon, S., Birchfield, S.: Training deep networks with synthetic data: Bridging the reality gap by domain randomization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 969–977 (2018)
55. Turiel, A., Parga, N., Ruderman, D.L., Cronin, T.W.: Multiscaling and information content of natural color images. Physical Review E **62**(1), 1138 (2000)
56. Ulyanov, D., Vedaldi, A., Lempitsky, V.: Deep image prior. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 9446–9454 (2018)
57. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7794–7803 (2018)
58. Wang, Y., Huang, H., Xu, Q., Liu, J., Liu, Y., Wang, J.: Practical deep raw image denoising on mobile devices. In: European Conference on Computer Vision. pp. 1–16. Springer (2020)

59. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing **13**(4), 600–612 (2004)
60. Wei, K., Fu, Y., Yang, J., Huang, H.: A physics-based noise formation model for extreme low-light raw denoising. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2758–2767 (2020)
61. Wei, K., Fu, Y., Zheng, Y., Yang, J.: Physics-based noise modeling for extreme low-light photography. IEEE Transactions on Pattern Analysis and Machine Intelligence (2021)
62. Zhang, K., Zuo, W., Chen, Y., Meng, D., Zhang, L.: Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. IEEE Transactions on Image Processing **26**(7), 3142–3155 (2017)
63. Zhang, K., Zuo, W., Zhang, L.: Ffdnet: Toward a fast and flexible solution for cnn-based image denoising. IEEE Transactions on Image Processing **27**(9), 4608–4622 (2018)
64. Zhang, K., Zuo, W., Zhang, L.: Learning a single convolutional super-resolution network for multiple degradations. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3262–3271 (2018)
65. Zhang, Y., Tian, Y., Kong, Y., Zhong, B., Fu, Y.: Residual dense network for image super-resolution. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2472–2481 (2018)
66. Zhou, Y., Jiao, J., Huang, H., Wang, Y., Wang, J., Shi, H., Huang, T.: When awgn-based denoiser meets real noises. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 13074–13081 (2020)

A Practical estimation of noise parameters for RAW images

In this appendix, we will go over the details of our estimation of the noise parameters for two different RAW images datasets : the SIDD dataset [1], and the See-in-the-Dark dataset [14].

A.1 SIDD dataset noise parameters

The SIDD dataset is made of RAW images taken with 5 different phones (Iphone 7, Samsung S6, Google Pixel 3, LG 4, Nexus 6), at different ISOs in different lighting conditions. Each image comes with a set of noise parameters, but as we mentioned earlier, this parameters are not accurate. Therefore, we estimate the noise parameters for each existing (phone, ISO) pair in the original dataset.

For each (phone, ISO) pair (P, I), we collect all the corresponding images in the dataset in a subset $D_{P,I}$. For each image in $D_{P,I}$ we estimate 2 noise parameters (K, σ) per channel. Our estimation method is based on simple properties of the Poisson law. Let us consider X a clean reference, and Y its noisy counterpart. Following our noise modeling, $Y \sim K \times \mathcal{P}(X/K) + \mathcal{N}(0, \sigma^2)$. Therefore $\mathbb{E}[Y] = X$ and $Var(Y) = KX + \sigma^2$, so that there is an affine relationship between the variance and the expectation of the noisy observation. A regression model is then used to estimate those parameters. Now, given a clean and homogeneous patch $\mathbf{X}$ of grey level u, we can compute the empirical variance v of the corresponding noisy patch $\mathbf{Y}$. The pair (u, v) should follow the affine relationship. This reasoning motivates our estimation procedure.

Algorithm 2: Noise parameter estimation for a (P,I) pair

Input : $D_{P,I}$
Output: $\hat{K}, \hat{\sigma^2}$
$K, \sigma^2 =$ `list(),list()`
for $(X, Y) \in D_{P,I}$ **do**
  `p_clean,p_noisy` $=$ `EXTRACT_PATCHES`$(X, Y, 11)$
  `m_clean,v_clean` $=$ `STATS(p_clean)`
  `m_noisy,v_noisy` $=$ `STATS(p_noisy)`
  `ind` $=$ `argsort(v_clean)`$[0 : N_5]$
  $U, V =$ `m_clean(ind),v_noisy(ind)`
  `model` $=$ `Theil_Sen_Regressor.fit`(U, V)
  if *model.score()* > 0.7 **then**
    K`.append(model.slope)`
    σ^2`.append(model.intercept)`
  end
end
$\hat{K} =$ `median`(K)
$\hat{\sigma^2} =$ `median`(σ^2)

First, we extract all the patches of size $(11, 11)$ of a clean image $\mathbf{X}$ and a noisy image $\mathbf{Y}$. For each patch, we compute its mean and variance. We select the 5% of the clean patches with the lowest variance, in order to select the ones which are the most homogeneous. We form a first vector U containing the mean of those patches in the clean image. We them form a second vector V containing the variance of the same patches in the noisy image. Thanks to the pair (U, V) we can fit a Theil–Sen regressor to find the slope K and the intercept σ^2 of the affine line. We chose this regression method because it is more robust to outliers than the linear regressor. If the regression fails to reach a satisfactory score of 0.7, we discard this parameter estimation. In Figure 11, we show an example of the graph (U, V) and its regression for an image at ISO 3200 for the Samsung S6.

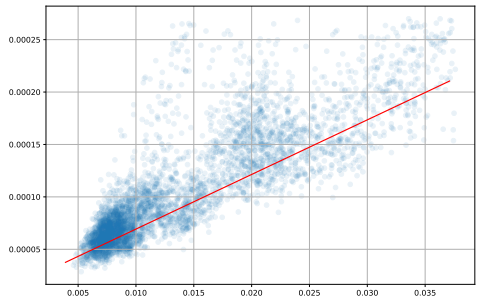


Fig. 11: In blue, a scattering plot of the local variance against the local mean. In red, the Theil-Sen regression of these points.

This method is applied for each image of the $D_{P,I}$, yielding a large number of estimates for K and σ^2. To give a unique estimate of these parameters, we select the median of all the estimates for each parameter. In the following Figure 12, we plot the estimated parameters in a logarithmic scale. Interestingly, the parameters seem to follow an affine rule, as was noted in [7].

To generate a noisy image from a clean RAW image, a small but important detail needs to be clarified. All images in the SIDD dataset are normalized between 0 and 1. However the Poisson noise is defined for integer values. Because clean RAW images are obtained by averaging 400 noisy pictures, the clean images have an artificial float precision. To generate Poisson noise, we first quantize the clean RAW image on N bits, where N depends on the camera. We then add Poisson noise, before normalizing and adding the read noise. The simulated noise is very close to the real one as we can see in Figure 13.

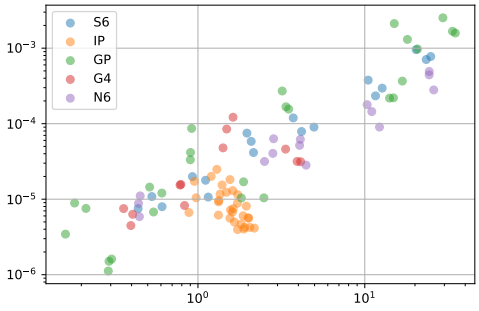


Fig. 12: Noise parameters of the SIDD dataset. We plot the read noise parameter σ^2 against the shot noise parameter K for each phone at different ISOs. The different colors represent different phones.

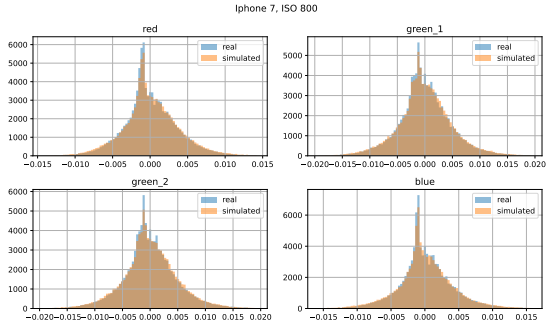


Fig. 13: Histograms per channel of a comparison of the difference between a clean and a noisy image, whether it is a simulation or a real noisy image. The exemple image was shot at ISO 3200 with the Samsung S6.

A.2 See-in-the-Dark noise parameters

For this task, we recall that the noise model is more complex than in the previous paragraph and is given by

$$Y = X + N_s(X) + N_r + N_b + N_q,$$

where $N_s(X)$ accounts for the shot noise, N_r for the read noise, N_b for the banding pattern noise, and N_q for the quantization noise. In order to improve its dynamic, the noisy image is multiplied by a factor γ of the order of 10^2 before being fed into the network. This factor γ corresponds to the ratio of the exposure times in the original See-in-the-Dark dataset. This amplification of the dynamic makes it necessary to have a good accuracy of the noise model and a very high precision in the estimation of its parameters, for all considered ISOs.

Some of these parameters are provided by the website "Photons To Photo"[3]: the read noise variance σ^2 at different ISOs and the shot noise parameter K only for ISO 100. Assuming a linear relationship between the parameter K and the ISO, this value can be extrapolated to other ISOs. In the following, we explain how to estimate all parameters. These parameters are then compared to those available on the "Photons to Photo" when available, as well as to the linear extrapolation for K.

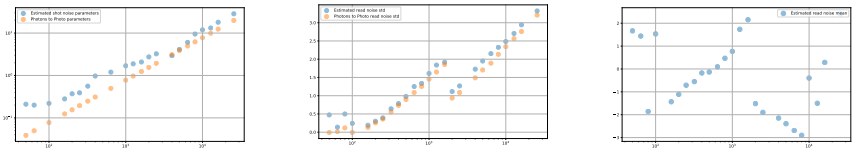
The shot noise parameter estimation algorithm is very similar to the one we used for the SIDD images, but the specific format of the See-in-the-Dark images and the particularity of their dynamic range should be taken into account. Indeed, these RAW images are coded in 14 bits integers for the Sony α7s2 camera where the black level is set to 512 and the highest value is set to 16384. We first subtract the black level to both the reference and the noisy image. Then, we divide the reference image by the exposure time ratio, to bring it to the dynamic of the short exposure image. We then run our estimation algorithm for both the digital gain K and the read noise variance σ^2, as presented in the previous section.

We report in Figure 14a and Figure 14b the estimates of the shot noise gain K and the read noise variance σ^2. As far as the read noise parameter σ^2 is concerned, we see that our estimates are very close to the one provided by the website, with a small discrepancy at lower ISOs. For the shot noise parameter K our estimates differs a little from the extrapolated values of the website.

In Figure 14c we report our estimates of the read noise mean. For each pair of noisy and clean images at a fixed ISO, we compute the difference of the noisy image and the rescaled clean image. We then compute the mean of this error which gives an estimate of the read noise mean at each ISO.

To estimate the banding noise parameter for each ISO, we perform similarly as for the read and shot noise parameter estimation. We select (50,50) patches of low variance in the clean and noisy images with the lowest possible intensity. Then we compute the mean for each row. This value corresponds to the mean value of the signal plus the offset added by the banding noise. We therefore com-

[3] https://www.photonstophotos.net/



(a) Shot noise parameter estimate $\hat{K}$ (b) Read noise variance estimate $\hat{\sigma^2}$ (c) Read noise mean estimate $\hat{\alpha}$

Fig. 14: Estimation of our shot noise and read noise parameters at different ISO. Comparison of our estimates against Photons to Photo's

pute the variance of these estimations of the mean, in order to give an estimate of the banding noise variance. We report our estimates of this parameters at different ISOs in Figure 15.

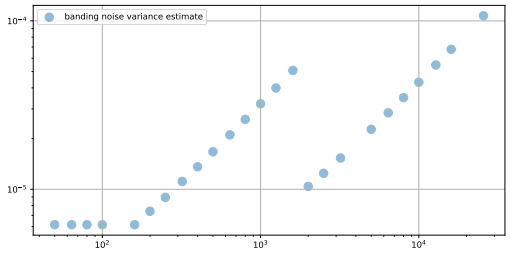


Fig. 15: Read noise parameter estimations at different ISOS.