DOI: 10.2352/ISSN.2470-1173.2016.18.DPMI-015

Local denoising applied to RAW images may outperform non-local patch-based methods applied to the camera output

Gabriela Ghimpeţeanu*[1]*, Thomas Batard*[1]*, Tamara Seybold*[2] *and Marcelo Bertalmío*[1]*;
[1] ***Information and Communication Technologies Department, Universitat Pompeu Fabra, Barcelona, Spain***
[2] ***Arnold & Richter Cine Technik (ARRI), München, Germany***

Abstract

State-of-the-art denoising methods achieve impressive results, even for large noise levels. However, they can not be implemented in camera hardware, mainly due to the fact that they are computationally too intensive. The aim of this paper is then to show that we can obtain comparable denoising results to the ones obtained with state-of-art methods by inserting a well-chosen fast denoising method at the right location in the camera processing pipeline. We evaluate our results visually and with respect to objective measures.

Introduction

Real-world scene camera acquisition generates noise due to physical and technological limitations, therefore denoising is performed at some stage of the formation of the output image. Nevertheless, the output image can still contain noise, especially if the photo is not taken with optimal camera parameters, or if the scene lighting conditions are challenging. Hence, there is still room for improvement in the denoising carried out in-camera in the image processing pipeline.

Camera makers do not usually provide information about this pipeline, but some in-camera denoising techniques are well established, e.g. correlated double sampling, consisting in sampling two images, one with the shutter closed and another after exposure, and subtracting the latter from the former, thus reducing *dark current* noise; another common in-camera denoising method is *coring*, the thresholding of the DCT coefficients corresponding to high spatial frequency information that is usually associated with noise. What all denoising and any other in-camera processes must have in common is a low computational complexity, and a very good compromise between the processing power they require and the visual quality of the results they provide; we refer the reader to [3] for more details about the camera processing pipeline.

On the other hand, over the last three decades, image denoising has been widely investigated and several approaches have been proposed. However, the methods developed mainly privilege the quality of the denoising, while ignoring the feasibility of their implementation in cameras. The first important class of denoising methods that appeared in the literature is the class of the so-called "local methods", referring to the fact these methods modify pixels based on the values of their neighbors, either through a single shot procedure (e.g. convolution with a kernel) or through an iterative procedure (gradient descent of an energy). Local methods are simple and can be implemented in cameras. However, local methods do not separate well the noise from the edges and textures and the noise removal makes the edges and textures be oversmoothed. The best local methods are derived from the Rudin-Osher-Fatemi (ROF) model [13] based on the reduction of the Total Variation (TV) of an image. More recently, a breakthrough has been made simultaneously by the Non-Local Means (NLM) algorithm of Buades et al. [4] and the approach of Awate et al. [2] that perform denoising through the averaging of image patches all over the image domain. The efficiency of those "non local methods" to denoise natural images is due to the self-similarity of patches on natural images. Finally, the state-of-the-art denoising methods are derived from the Block-Matching and 3D Filtering (BM3D) algorithm [8] that combines patch-based approaches with frequency filtering in a non trivial way, outperforming the aforementioned patch-based methods in a great extent. We refer to [9] for a complete introduction to the denoising problem.

Denoising can be applied at different stages in the camera processing pipeline. It can be applied on the RAW data, in which, however, the neighboring pixels are of different color; usually difficult for standard denoising algorithms. Alternatively, denoising can be applied on the monitor-ready image data. However, noise characteristics are extremely complex after all processing steps, see [14] for details. Due to the complex noise characteristics, the success of state-of-the-art denoising algorithms, tested on the usual image sets, can clearly drop on real camera data. In this paper, we therefore propose applying denoising at an earlier stage of the camera processing pipeline.

State-of-the-art denoising methods, being based on patch comparisons in the image domain, are computationally too intensive for camera implementation. The aim of this paper is to propose a strategy to overcome this drawback. We show that applying a standard local method at some stage in the camera processing pipeline can provide better results than applying a non-local patch-based method to the output of the camera processing pipeline. More precisely, we show that, for noise levels that are not too large, the proposed strategy can produce more pleasant images and better results with respect to the PSNR and SSIM metrics. For a given camera, our strategy requires the simulation of its processing pipeline, as camera makers do not provide all the information on the pipelines their cameras are using. Inserting the denoising method at different stages of the (simulated) pipeline, and testing the quality of the corresponding camera output denoised images, we show that the best results are obtained when the denoising method is applied early in the pipeline on the RAW demosaicked image.

The outline of the article is the following. In Section 2, we present the local and the non-local patch-based denoising methods we will use in our experiments. The proposed strategy as well as experiments on a standard digital camera are presented in Section 3.

DOI: 10.2352/ISSN.2470-1173.2016.18.DPMI-015

Finally, in Section 4, we discuss our current works: adding an extra step in the camera processing pipeline in order to improve our strategy and extend it to other types of cameras.

Implementation of the denoising method inserted in the camera processing pipeline

In the experiments we perform in Section 3, the local denoising method we insert in the camera processing pipeline is the ROF model. This model has been widely investigated both theoretically and numerically over the last two decades, and we refer the reader to [7] for a thorough analysis of that model. In what follows, we provide the necessary information for the understanding of our implementation of the model.

The original formulation of the ROF denoising model is the following. We assume that the observed gray-level image I_0 is the result of the corruption of a clean image I_{clean} with additive white Gaussian noise of standard deviation σ, i.e.

$$I_0 = I_{clean} + n$$

where

$$\frac{1}{|\Omega|}\int_\Omega n^2\, d\Omega = \sigma^2$$

and Ω denotes the image domain.

The ROF model aims at recovering I_{clean}, through the following variational formulation

$$\arg\min_I \int_\Omega \|\nabla I\|\, d\Omega \quad s.t.\ \frac{1}{|\Omega|}\int_\Omega (I-I_0)^2\, d\Omega = \sigma^2 \tag{1}$$

which is equivalent to

$$\arg\min_I E(I) :\ = \int_\Omega \lambda (I-I_0)^2 + \|\nabla I\|\, d\Omega \tag{2}$$

where λ is a Lagrange multiplier. The term

$$\int_\Omega \|\nabla I\|\, d\Omega$$

is called the Total Variation (TV) of I. It can be shown that the problem (2) has a unique solution (see e.g. [7] for a proof).

The ROF model is not differentiable, meaning that gradient descent-based algorithms can not be applied to solve it numerically. The approach used originally by Rudin et al. [13] consisted in considering a differentiable energy E_ε approximating E, and the corresponding variational model

$$\arg\min_I E_\varepsilon(I) :\ = \int_\Omega \lambda (I-I_0)^2 + \sqrt{\|\nabla I\|^2 + \varepsilon} \quad d\Omega \tag{3}$$

for ε very small, that can be solved through a gradient descent algorithm.

The gradient of the energy E_ε at I is

$$\nabla E_\varepsilon(I) = \lambda (I-I_0) + \nabla^* \left(\frac{\nabla I}{\sqrt{\|\nabla I\|^2 + \varepsilon}} \right) \tag{4}$$

where ∇^* is the adjoint of the gradient operator ∇, and the authors performed the corresponding gradient descent

$$I_{t+dt} = I_t - dt\, \nabla E_\varepsilon(I_t), \qquad I_{|t=0} = I_0 \tag{5}$$

until reaching the steady-state.

In (5), the scalar λ, considered as a Lagrange multiplier associated to the noise level, is updated at each iteration.

As we will see in the following Section, the situation is different in our context as the noise statistics are not known, meaning that we can not directly use the original ROF denoising model (1) or its differentiable approximation (3) since the constraint

$$\frac{1}{|\Omega|}\int_\Omega (I-I_0)^2 d\Omega = \sigma^2$$

assumes that the noise is Gaussian and its variance is known. Our proposal is then to consider the gradient descent associated to the differentiable approximation of the unconstrained ROF model, i.e.

$$I_{t+dt} = I_t - dt\, \nabla^* \left(\frac{\nabla I}{\sqrt{\|\nabla I\|^2 + \varepsilon}} \right) \tag{6}$$

and stop the iteration at some point, that will be determined in the following Section.

Finally, let us point out that more sophisticated algorithms have been developed since the original one (5) to solve the initial problem (1). For instance, Chambolle [6] proposed an algorithm based on the dual formulation of (1), and Zhu et al. [16] on the primal-dual formulation of (1).

Experiments on a standard camera

Simulating the camera processing pipeline

The demonstration of our claim requires the reproduction of the image processing pipeline of a camera, as we aim at inserting a denoising method at some stage of the pipeline and evaluate the output denoised image.

The basic steps of the pipeline, common to every standard camera, are the following:

1. **Recording** In this paper, we are making our experiments with the Nikon D3100 camera. A photo taken with this camera is captured in the RAW format NEF of Nikon cameras, obtaining the CFA (color filter array) RAW data with a Bayern mosaic pattern. It is a 12-bit depth image.
2. **White-balance** Afterwards, a white-balance step is applied, which assures that the image has no color cast. This is done by scaling all intensity values with parameters read from the RAW file, such that neutral colors keep a correct appearance.
3. **Demosaicking** The camera sensor produces an image in which for each pixel we only get one of the image channel intensity values (either red, or green or blue), and we need to find an estimate of the other two missing values. This is done by an interpolation process called demosaicking, which produces an image with 3 channels. In the next Section, we perform our experiments on denoising by testing the following two different (local) demosaicking algorithms.

   The first one, based on bilinear interpolation, is one of the simplest demosaicking methods, where the missing value of a certain color channel is calculated as the average of the four neighbour pixels of the same color channel.

   The second one, proposed by Malvar et al. [12], is based

DOI: 10.2352/ISSN.2470-1173.2016.18.DPMI-015

on bilinear interpolation and further refined by using the correlation among the RGB channels, with Laplacian cross-channel corrections.

4. **Color Correction** After demosaicking, a color correction step is applied, which converts the image from the camera color space to sRGB (standard RGB color space).
5. **Gamma Correction** Then, a gamma correction is performed, with the standard value of 1/2.2.
6. **Quantization** The final step of the pipeline (for our purposes) consists in quantizing the image from 12-bit depth to 8-bit depth, providing an RGB image ready for display.

There are more steps, like contrast enhancement or compression, but we omit them for simplicity. More details about the cameras processing pipeline of standard cameras can be found in [3].

Denoising experiments

For tests on color images we proceed as follows.

1. Take a clean RAW image C_{RAW} and add white Gaussian noise of standard deviation σ to it to create the noisy RAW image N_{RAW}. Apply the previously described color processing pipeline on the RAW 12-bit depth images C_{RAW} and N_{RAW}, obtaining as output the 8-bit depth *RGB* images C_{RGB} and N_{RGB} respectively.
2. Apply a non-local patch-based denoising method on the 8-bit depth noisy image N_{RGB}, obtaining the denoised image NL_{RGB}.
3. Apply white balance and demosaicking on N_{RAW}, and then denoise this 12-bit depth image with a local denoising method, obtaining the denoised image L_{RAW}. It is worth noting that applying the demosaicking on N_{RAW} makes the noise be not Gaussian any more, and this is why we use the unconstrained version (6) of the ROF model to denoise the image. Afterwards, apply the rest of the same color processing pipeline on the image L_{RAW} to get the final 8-bit depth *RGB* image L_{RAWRGB}.
4. Compute the PSNR and SSIM index values of NL_{RGB} and L_{RAWRGB} with respect to the ground truth C_{RGB}.

Experiments were done on a collection of 20 color images that we captured with the Nikon D3100 camera. We tested 6 different noise levels σ. The local denoising method applied at the Step 3 is the iterative algorithm (6) that we apply channel-wise. Given a noise level σ, we stop the algorithm after the same number of iterations for each image of the dataset, i.e. we choose the one that maximizes the average PNSR value of the images L_{RAWRGB} over the dataset. The non-local patch-based methods that we apply at the Step 2 on the image N_{RGB} are the NLM [4] and BM3D [8] algorithms, whose implementations are available online [5], respectively [11]. The standard parameters of these models are determined by the standard deviation of the noise assuming that this latter is Gaussian, which is not the case for the images N_{RGB}, and therefore we have to tune the parameters manually. As we did for the local method, given a noise level σ, we choose the same parameter for each image, choosing the one that maximizes the average PSNR value of NL_{RGB} over the database.

The left plot in Fig.1 illustrates the average PSNR values over our proposed image dataset, for each noise level and each denoising strategy aforementioned. For the color processing pipeline we simulate, we applied the local demosaicking algorithm of Malvar, He, and Cutler [12]. The right plot in Fig.1 shows the corresponding average SSIM index values. We compute the SSIM index of a color image as the mean of the SSIM indices of the channels. We can see that, up to a certain noise level, our denoising method described previously in Step 3 produces better results in terms of PSNR and SSIM index than the two non-local methods NLM and BM3D. For higher noise level, denoising with a non-local method like BM3D gains an advantage.

The plots show on the horizontal axis the noise standard deviation added to the RAW σ_{RAW}, computed as equivalent values for an 8-bit depth image. As the RAW images are flat, the added noise variance values are very small. We consider 6 noise levels. After applying all the steps in the color processing pipeline, the variance increases, with an increment that varies for each image. For example, for the first noise level considered, we add Gaussian white noise with standard deviation of value 1.14 in a 0-255 scale image. After applying the color processing pipeline and estimate the noise level for each 8 bit- depth reconstructed RGB image, the values of the standard deviation vary between 4 and 8. Although the same noise variance value is used for adding noise to the RAW, the final RGB noise level can vary from one image to another. Table 1 illustrates, for each of the 6 noise levels, the variance values added to the RAW, and the corresponding average standard deviation σ_{avg} in the final RGB. The value of σ_{avg} is computed as the average of all standard deviation values for the final noisy RGB images from our data base.

Standard deviation σ_{RAW} of noise added to the RAW, and the corresponding σ_{avg} on the final RGB, as average over our data base

σ_{RAW}	1.14	1.80	2.55	3.12	4.41	5.1
σ_{avg}	6.12	9.22	12.36	14.57	19.04	21.16

A visual comparison is illustrated in Fig.2, where the images are denoised with the optimal parameters described above. In the crop images from the first row, we can see that our method preserves better the details of the green wool thread, compared to NLM and BM3D. In the second row, BM3D produces a denoised image with a higher PSNR than our method result, for a higher noise level. However, we can see that our image result preserves slightly better the sharpness of the text compared to BM3D. For the images from the third row, our denoised image gives a better reconstruction of the red flowers and a sharper leaf texture, compared to the other two denoising methods. In the last row, our denoised image preserves better the stone texture compared to BM3D and NLM.

We claim that the advantage of our proposed strategy, that consists in denoising the demosaicked RAW image N_{RAW} with a local method and then apply the rest of the camera processing pipeline, is due to the fact that the local denoising method is applied before both the gamma correction and quantization steps. This affirmation is sustained by the following two experiments.

The aim of the first experiment is to show the disadvantage for the local denoising method to be applied after quantization. To that purpose, we modify the simulated color processing pipeline. The steps are: 1.Recording; 2.White-balance; 3.Demosaicking;

DOI: 10.2352/ISSN.2470-1173.2016.18.DPMI-015

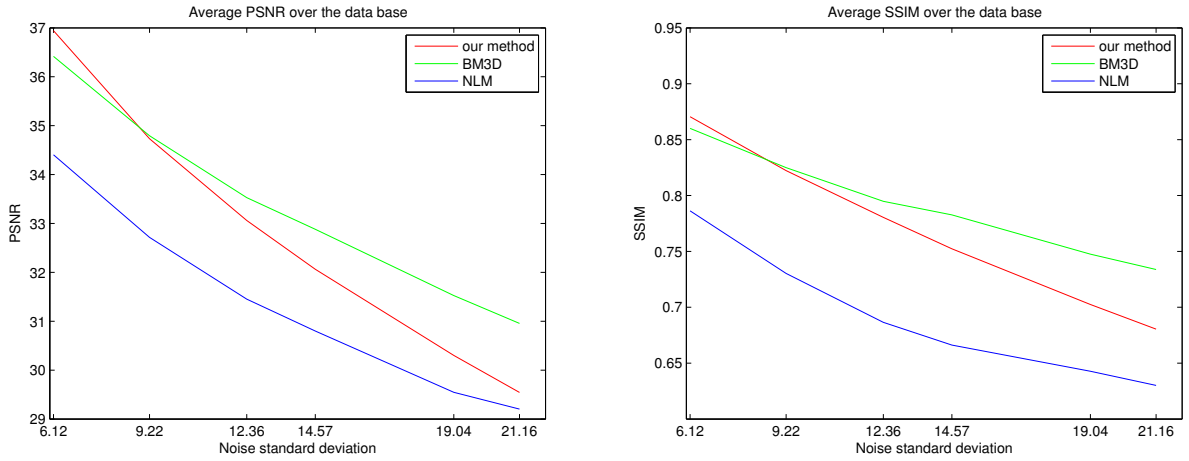


Figure 1. *Average PSNR value (left) and SSIM index value (right) computed over our proposed image data base, for comparing our proposed local denoising method to BM3D and NLM denoising.*

4.Quantization from 12-bit depth to 8-bit depth; 5.Color Correction; 6.Gamma Correction. Hereby, we apply after Step 4 our local denoising method (6) to an 8-bit depth image. Followed by Steps 5 and 6, we get the denoised image L_{RAWRGB}. We apply the non-local denosing algorithm after the Step 6, and obtain the denoised image NL_{RGB}. We perform this experiment on 4 images from our image database, where the non-local patch-based denoising method considered is the BM3D algorithm. Given a noise level σ, the parameters of each denoising method are optimized for each image according to the PSNR values, both for L_{RAWRGB} and NL_{RGB}. The left plot in Fig.3 shows these PSNR values for all the noise levels considered. Unlike the previous experiment, we observe that the results are now in favour of the strategy that consists in applying a non-local patch-based method to the camera noisy output image.

In order to show the disadvantage for the local denoising method to be applied after gamma correction, we do not modify the simulated color processing pipeline. The chain is: 1.Recording; 2.White-balance; 3.Demosaicking; 4.Color Correction; 5.Gamma Correction; 6.Quantization from 12-bit depth to 8-bit depth. Hereby, we modify the location at which we apply our denoising method (6) in the pipeline: after Step 5. Followed by Step 6, we obtain the denoised image L_{RAWRGB}. Again, we compare this image to the result of applying the non-local denosing algorithm at the end of the pipeline, and obtain the denoised image NL_{RGB}. The parameters of the denoising methods have been optimized as it was done in the previous experiment. The right plot in Fig.3 shows the average PSNR values of the subsequent images L_{RAWRGB} and NL_{RGB} over the same set of 4 images aforementioned, for different noise levels σ, where the non-local patch-based method is BM3D. As in the previous case, we observe that the results are in favour of the strategy that consists in applying a non-local patch-based method to the camera noisy output image.

Current work

1. Applying a non-linear function before denoising the RAW

Current work is devoted to improving the local denoising method that we apply to the demosaicked RAW.

We have seen in one experiment, described in the previous Section, that applying local denoising after the gamma correction step in the color processing pipeline gives an image of low quality. The PSNR comparison between local ROF and non-local BM3D denoising for this test is illustrated in the right plot from Fig.3. However, experiments showed that for some images, applying a non-linear function before denoising, and the inverse of it afterwards, can improve the denoising result. The experiments are done as in the following.

1. Take a clean RAW image C_{RAW} and add white Gaussian noise of standard deviation σ to create the noisy RAW image N_{RAW}. Apply the previously described color processing pipeline on the 12-bit depth RAW images C_{RAW} and N_{RAW}, obtaining as output the 8-bit depth *RGB* images C_{RGB} and N_{RGB} respectively.
2. Apply white balance and demosaicking on N_{RAW}, and then denoise this 12-bit depth image with a local denoising method, obtaining the denoised image L_{RAW}. Afterwards, apply the rest of the same color processing pipeline on the image L_{RAW} to get the final 8-bit depth *RGB* image L_{RAWRGB}.
3. Apply white balance and demosaicking on N_{RAW}, followed by a non-linear function like a gamma power, obtaining the noisy image N_γ. We denoise this 12-bit depth image with a local denoising method, obtaining the denoised image L_γ. We apply the inverse of the non-linear function, followed by the rest of the color processing pipeline, getting the final 8-bit depth *RGB* image $L_{\gamma RGB}$.
4. Compute the PSNR of L_{RAWRGB} and $L_{\gamma RGB}$ with respect to the ground truth C_{RGB}.

The local denoising method applied here is the same iterative algorithm (6) that we apply channel-wise. Current work shows that

DOI: 10.2352/ISSN.2470-1173.2016.18.DPMI-015

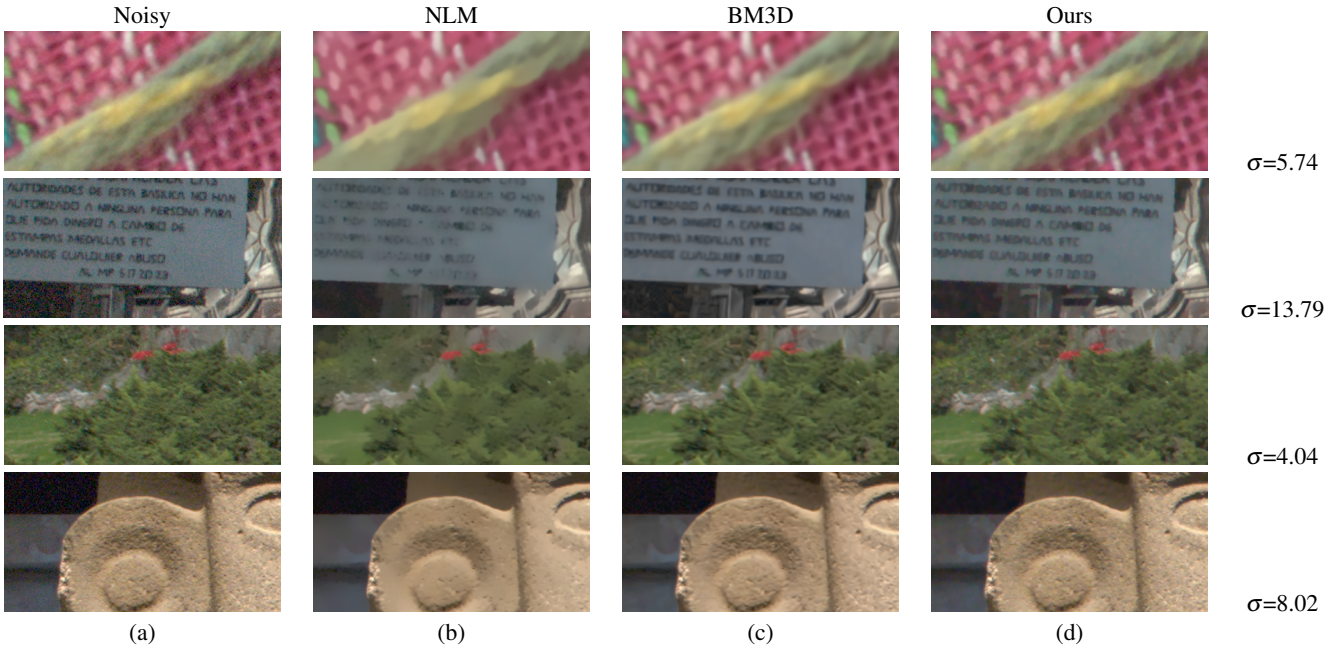


Figure 2. *Comparison of our local denoising framework to NLM and BM3D denoising algorithms. Row 1. (a) crop from noisy image "image1" with $\sigma = 5.74$. (b) NLM result, PSNR=33.06. (c) BM3D result, PSNR=35.52. (d) our result of applying TV to the demosaicked RAW image, PSNR=35.77. Row 2. (a) crop from noisy image "image11" with $\sigma = 13.79$. (b) NLM result, PSNR=31.37. (c) BM3D result, PSNR=33.63. (d) our result of applying TV to the demosaicked RAW image, PSNR=33.43. Row 3. (a) crop from noisy image "image12" with $\sigma = 4.04$. (b) NLM result, PSNR=33.54. (c) BM3D result, PSNR=36.37. (d) our result of applying TV to the demosaicked RAW image, PSNR=37.50. Row 4. (a) crop from noisy image "image20" with $\sigma = 8.02$. (b) NLM result, PSNR=30.81. (c) BM3D result, PSNR=31.35. (d) our result of applying TV to the demosaicked RAW image, PSNR=32.27.*

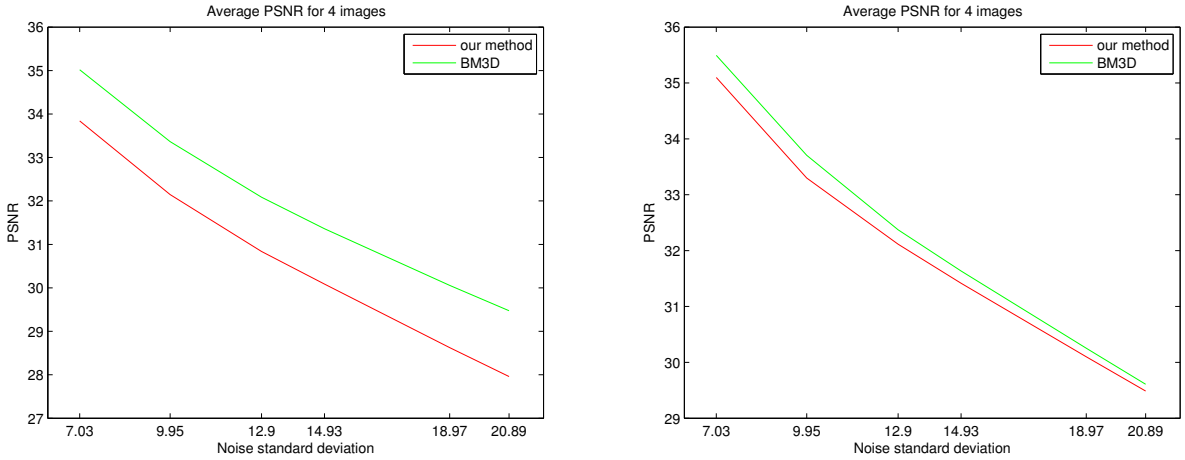


Figure 3. *Left: Average PSNR value for 4 images from our data base, for comparing our proposed local denoising method on a 8-bit depth image to BM3D. Right: Average PSNR value for 4 images from our data base, for comparing our proposed local denoising method applied after the gamma-correction to BM3D.*

applying a non-linear function before the denoising step can bring an improvement. The challenge here is how to choose the optimum non-linear function for each image. For example, when using a gamma power function, the optimum value for gamma differs from image to image. This value can be optimized according to the PSNR index value.

Fig.4 illustrates a current work image example where we compare our method result to that of applying a gamma power function before our denoising method. A comparison is also done with respect to the BM3D result. On the first two rows cropped from "image1", we can see that by applying the non-linearity, we obtain an image with a less noisy background, that preserves the sharpness of the orange or blue wool thread, compared to our method without a non-linearity and to BM3D. Another advantage of using

DOI: 10.2352/ISSN.2470-1173.2016.18.DPMI-015

the non-linear function can be seen in the plots in Fig.5. For the same test image, the noise level up to which our local method is better, in terms of PSNR, than BM3D is increased.

2. Experiments on ARRI images

Camera processing pipeline

The ARRI ALEXA cameras are used for digital cinema acquisition. The pipeline for processing *ARRI* images [1] from RAW to a final RGB image is different than the Nikon one: it starts with the linear ARRI RAW image of 16-bit depth, followed by white balance and demosaicking. Afterwards, a color correction matrix is applied, which outputs an image in a wide-gamut color space. A logarithm-based non-linear function named Log C is applied on this data, followed by a tone-mapping curve that mimics a motion-picture print film effect, responsible for adjusting the contrast and the details. Another matrix is then used on the tone-mapped data to make the transition to the sRGB color space Rec709. Finally, a power function with an exponent value of 1/2.4 makes the data ready for monitor display. The ARRI color processing pipeline starts with a 16-bit depth RAW and ends with a 16-bit depth final RGB image.
In the middle of the process, some *ARRI* cameras have a step of quantizing to 10-bit depth the log C data. In this context, following the idea of the Nikon experiments and translating it to ARRI images, we make the following denoising experiment: apply a local denoising method to the demosaicked RAW image before quantizing to 10-bit depth, and compare this result to applying a non-local patch based method to the final 16-bit depth RGB image (obtained after a 10 bit quantization step). The local denosing method would have an advantage of 6-bit depth over the non-local one, fact that made the Nikon experiments (advantage of 4-bit depth) successful for moderate noise levels.
One possibility is to introduce the quantization step in the previously described ARRI color processing pipeline, after applying the log C curve. However, if a quantization to 10 bit is applied on the log C data, afterwards, in the color processing chain, the tone mapping step and the color correction would produce an 16-bit depth image that would reverse the 10 bit quantization advantage. This is why we use a general color processing chain for the ARRI images, as described in [1].
Our test starts with the same linear ARRI RAW image of 16-bit depth, followed by white balance and demosaicking. Afterwards, the RAW data is channel-wise normalized for the exposure, using the following formula for the red channel:

$$R = \frac{18EI}{400} \cdot \frac{R_{RAW} - 256}{65535 - 256}. \tag{7}$$

The same formula is applied for the channels G and B. The next step is a matrix multiplication that converts the data to a wide-gamut color space:

$$\begin{pmatrix} R_{wg} \\ G_{wg} \\ B_{wg} \end{pmatrix} = \begin{pmatrix} 1.1766 & -0.1190 & -0.0576 \\ -0.0194 & 1.0606 & -0.0412 \\ 0.0368 & -0.2019 & 1.1652 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}. \tag{8}$$

Afterwards, we make the conversion from this wide-gamut color space to the sRGB color space:

$$\begin{pmatrix} R_s \\ G_s \\ B_s \end{pmatrix} = \begin{pmatrix} 1.6175 & -0.5373 & -0.0802 \\ -0.0706 & 1.3346 & -0.2640 \\ -0.0211 & -0.2270 & 1.2481 \end{pmatrix} \begin{pmatrix} R_{wg} \\ G_{wg} \\ B_{wg} \end{pmatrix}. \tag{9}$$

After color correction, we apply a Log C function on the data and then quantize from 16 bit to 10 bit, following by applying the inverse Log C function.
The last step in the color processing chain is to apply channel-wise the correction power function:

$$C' = \begin{cases} 12.92C & if \quad C \leq 0.0031308 \\ 1.055C^{1/2.4} - 0.055 & if \quad C > 0.0031308 \end{cases} \tag{10}$$

where C is R_s, G_s or B_s. The output is a 16-bit depth image.

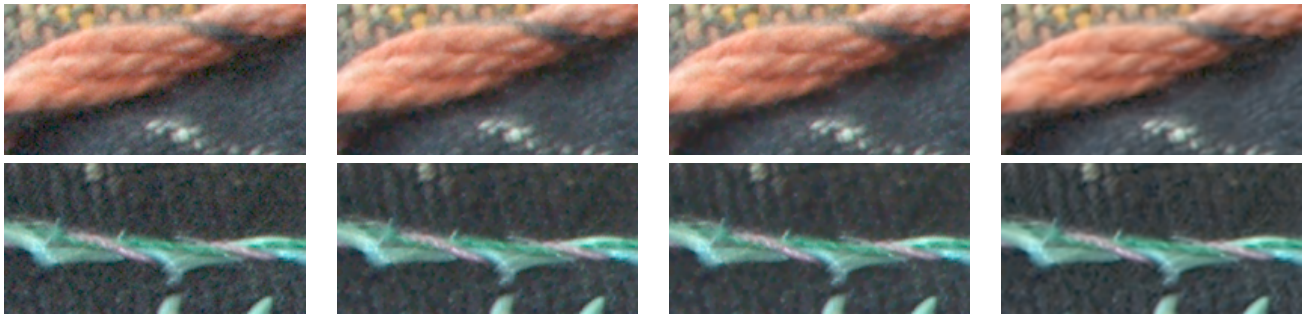
Denoising experiments

For tests on ARRI images, we have the noisy RAW image N_{RAW} and also the reference clean image C_{RAW}, obtained as the average of a series of noisy images taken in identical conditions. Starting with two 16-bit depth noisy and clean RAW images on which white balance and demosaicking are applied, we proceed as in the following:

1. Take the demosaicked and white-balanced noisy RAW ARRI image and apply the previously described basic color processing pipeline: normalization, conversion to wide-gamut color space, conversion to sRGB color space, Log C quantisation from 16 bit to 10 bit, applying the inverse of the Log C function and applying the correction power function. The output is the noisy image N_{RGB}. The same process is applied to the clean RAW image C_{RAW} getting the image C_{RGB}.
2. Apply a non-local denoising method on the 16-bit depth noisy image N_{RGB}, obtaining the denoised image NL_{RGB}.
3. Take the demosaicked and white-balanced noisy RAW ARRI image and do the normalization step. Afterwards apply a local denoising algorithm, followed by the rest of the basic ARRI pipeline: conversion to wide-gamut color space, conversion to sRGB color space, Log C, quantisation from 16 bit to 10 bit, applying the inverse of the Log C function and applying the correction power function. Obtain the denoised image L_{RAWRGB}.
4. Compute the PSNR and SSIM index values of NL_{RGB} and L_{RAWRGB} with respect to the ground truth C_{RGB}.

For the experiments, as a local denoising method we used ROF and as non-local denoising methods we used NLM and BM3D. Fig.6 illustrates a current work image example where the three denoising methods are compared, for which the denoising parameters are optimized according to the PSNR value. Although our method produces an image with a smaller PSNR that BM3D and NLM, the results are visually comparable, with the advantage that our method is simpler and faster. In tests with added Gaussian noise, the difference image between the noisy and the denoised image should show, ideally, uniformly distributed noise. However, for real noisy images, this is not necessarily the case, as the

DOI: 10.2352/ISSN.2470-1173.2016.18.DPMI-015



(a) (b) (c) (d)

Figure 4. *Comparison of our local denoising framework, with or without a non-linear function. Row 1 and 2. (a) crop from noisy image "image1". (b) our result, PSNR=35.77, no linear function. (c) our result, with a gamma power of 1.5, PSNR=36.15. (d) BM3D result, PSNR=35.51.*

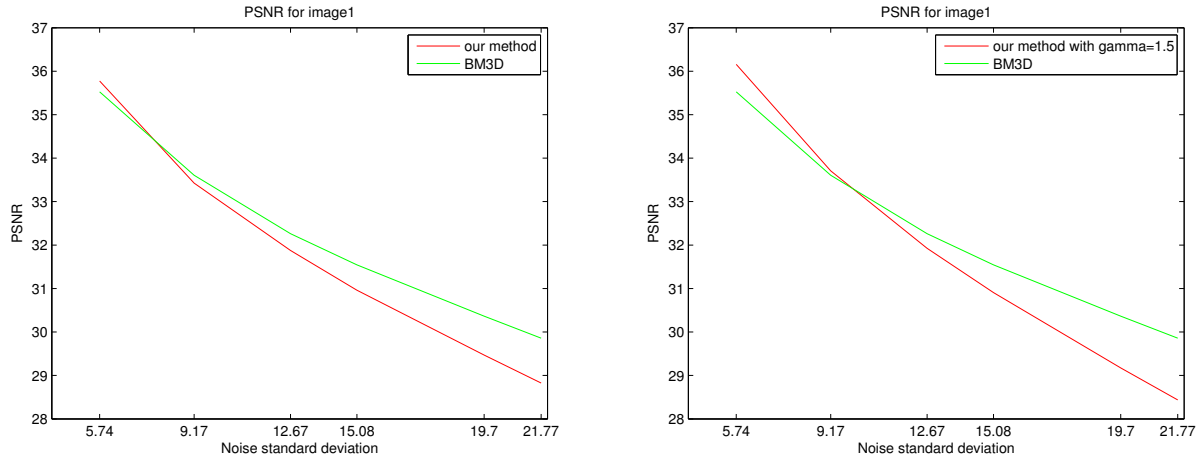


Figure 5. *Left: PSNR value for "image1", for comparing our proposed local denoising method to BM3D. Right: PSNR value for "image1", for comparing our local denoising method with a gamma power of 1.5 to BM3D.*

noise is signal-dependent. Denoising algorithms do better when the noise model is Gaussian, not realistic; when the noise is realistic, their denoising performance may decay significantly [14]. Fig.6 illustrates a real noisy image, with image-dependent noise, as we can notice in the difference images in Rows 2 and 4. We can see that our denoising result is comparable to the one of BM3D.

Conclusion

In this paper, we demonstrated the relevance of denoising a RAW image in the camera hardware with a fast local method rather than denoising the camera output image with a non-local patch-based method. We believe that this result may be of significance both in the academic research world and in the technology industry. To that purpose, we had to simulate the camera pipeline processing of a standard camera. We showed that the location in the camera processing pipeline in which we insert the local denoising method is crucial, i.e. the earlier in the pipeline the better. Besides pursuing the current work on cinema cameras and on the improvement of our local denoising method, we plan to analyse the limits of our approach, investigating the threshold of the noise level for which our local method outperforms non-local patch-based methods.

Acknowledgments

This work was supported by the European Research Council, Starting Grant ref. 306337, by the Spanish government, grant ref. TIN2012-38112, and by the Icrea Academia Award.

References

[1] S. Andriani, H. Brendel, T. Seybold, and J. Goldstone, Beyond kodak image set: A proposal for a new reference set of motion color images, ICIP, (2013).

[2] S.P. Awate and R.T. Whitaker, Higher-order image statistics for unsupervised, information-theoretic, adaptive, image filtering, Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit., 2, pg. 44-51. (2005).

[3] M. Bertalmío, Image Processing for Cinema, Boca Raton, FL: CRC Press, Taylor and Francis. (2014).

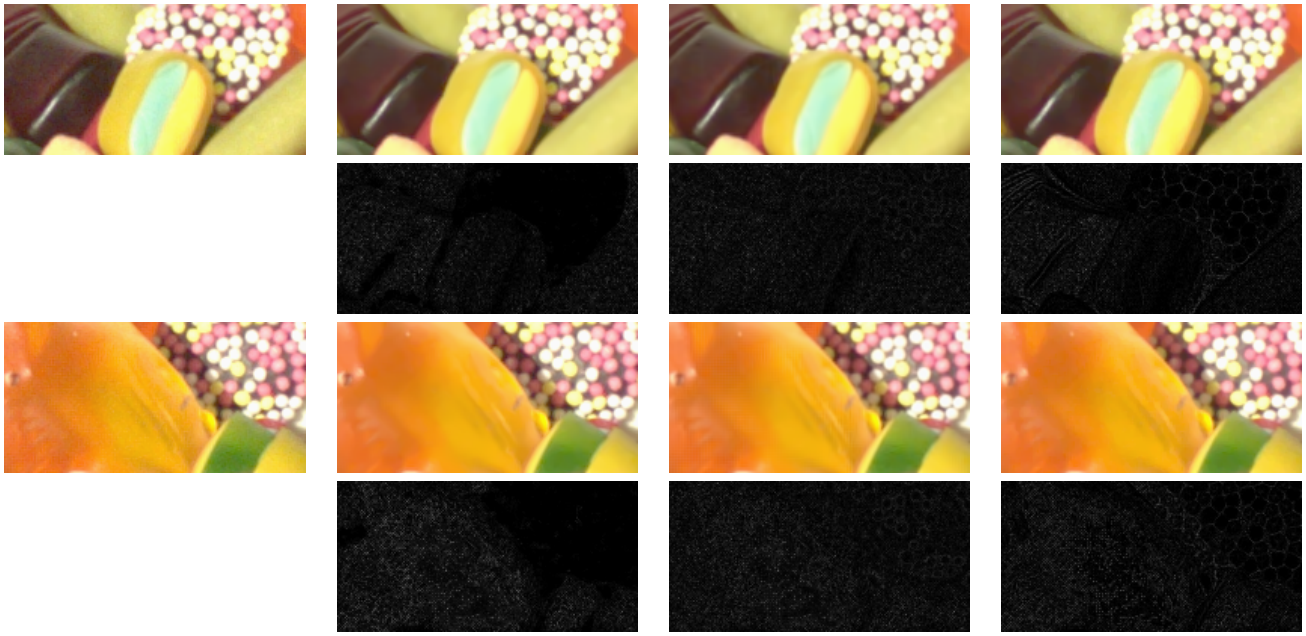
[4] A. Buades, B. Coll and J.-M. Morel, A non-local algorithm for image denoising, Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit., 2, pg. 60-65. (2005).

[5] A.Buades, B.Coll and J.-M.Morel, Non-local means denoising, Image Processing On Line, vol. 1, (2011).

[6] A. Chambolle, An algorithm for total variation minimization and applications, J. Math. Imaging Vis., 20, pg. 89-97. (2004).

[7] A. Chambolle, V. Caselles, D. Cremers, M. Novaga, and T. Pock, An

DOI: 10.2352/ISSN.2470-1173.2016.18.DPMI-015



(a) (b) (c) (d)

Figure 6. *Comparison of our local denoising framework to NLM and BM3D denoising algorithms, for the ARRI pipeline. Row 1 and 3. (a) crop from noisy ARRI image "candy". (b) NLM result, PSNR=40.72. (c) BM3D result, PSNR=41.18. (d) our result, PSNR=40.20. Row 2. Noise difference images for crops of Row 1, scaled for visualization with a scaling factor 7 (b) NLM result. (c) BM3D result. (d) our result. Row 4. Noise difference images for crops of Row 3, scaled for visualization with a scaling factor 7 (b) NLM result. (c) BM3D result. (d) our result.*

introduction to total variation for image analysis, Theoretical foundations and numerical methods for sparse recovery, 9, pg. 263-340. (2010).

[8] K. Dabov, A. Foi, V. Katkovnik and K. Egiazarian, Image denoising by sparse 3D transform-domain collaborative filtering, IEEE Trans. Image Process., 16(8), pg. 2080-2095. (2007).

[9] M. Lebrun, M. Colom, A. Buades and J.M. Morel, Secrets of image denoising cuisine, Acta Numerica, 21(1), pg. 475-576. (2012).

[10] M. Lebrun, M. Colom, J.M. Morel, The noise clinic: a universal blind denoising algorithm, IEEE Int. Conf. Im. Process., pg. 2674-2678. (2014).

[11] M. Lebrun, An analysis and implementation of the BM3D image denoising method, Image Processing On Line, vol. 2, pp.175-213, (2012).

[12] H. S. Malvar, L. He, R. Cutler, High-quality linear interpolation for demosaicing of Bayer-patterned color images, IEEE ICASSP, pg. 485-488. (2004).

[13] L.I. Rudin, S. Osher and E. Fatemi, Nonlinear total variation based noise removal algorithms, Physica D: Nonlinear Phenomena, 60(1-4), pg. 259-268. (1992).

[14] T. Seybold, Ö. Cakmak, C. Keimel and W. Stechele, Noise characteristics of a single sensor camera in digital color image processing, Color and Imaging Conference, pg. 53–58. (2014).

[15] Z. Wang, A.C. Bovik, H.R. Sheikh and E.P. Simoncelli, Image Quality Assessment: From Error Visibility to Structural Similarity, IEEE Trans. Im. Process., 13(4), pg. 600-612. (2004).

[16] M. Zhu and T. Chan, An efficient primal-dual hybrid gradient algorithm for total variation image restoration, CAM Reports 08-34, UCLA, Center for Applied Math. (2008)

Author Biography

Gabriela Ghimpețeanu *received the Diploma degree in Mathematics and Informatics at the University of Bucharest and the MSc in Visual Computing at Saarland University, Germany. In 2013 she has started the PhD in Image Processing for Enhanced Cinematography at Universitat Pompeu Fabra.*

Thomas Batard *received the MSc. degree in Mathematics and Computer Science (2006) and the PhD degree in Mathematics applied to Image Processing (2009) both in the University of La Rochelle, France. He was a post-doctoral fellow in the Department of Applied Mathematics of Tel Aviv University, Israel, in 2011. He also visited the Cinvestav Unidad Guadalajara, Mexico, as well as the Department of Signal, Image, Communications of the University of Poitiers, France, both for a period of 6 months. He has joined the University Pompeu Fabra in 2012 where he is working as a post-doctoral fellow in the group Image Processing for Enhanced Cinematography headed by Marcelo Bertalmo. His main research interests are variational methods and differential geometry and their applications to image processing and computer vision.*

Tamara Seybold *received the B.Sc., Dipl.-Ing. (M.Sc) and Dr.-Ing. degree in electrical engineering and information technology from the Technische Universität München (TUM), Munich, Germany in 2009, 2011 and 2015, respectively. She has been a Digital Imaging Scientist at Arnold & Richter Cine Technik (ARRI) since 2012.*

Marcelo Bertalmío *received the Ph.D. degree in electrical and computer engineering from the University of Minnesota in 2001. He is an Associate Professor at University Pompeu Fabra in Barcelona, Spain. His interests are Image Processing and Computer Vision for digital cinema applications, although he prefers the (analog) films of Luis Buñuel and Luis García Berlanga.*