Experimental report

Article 1: *An Analysis and Implementation of the BM3D Image Denoising Method,* Lebrun Marc

Article 2: *Exploring Patch Similarity in an Image*, Lisani, Jose-Luis and Morel, Jean-Michel

Article 3: *Implementation of the "Non-Local Bayes" (NL-Bayes) Image Denoising Algorithm*, Buades, Lebrun, Marc and Buades, Antoni and Morel, Jean-Michel

Compare set of patches

The first objective is an in-depth analysis of the self-similarity concept and the Gaussianity hypothesis in natural images. Indeed, for now, we have made 2 assumptions:

- Any random patch taken in a specific image will present spatially and visually similar patches.
- The similar patches follow a gaussian distribution.

To conduct those experiments, we will use 3 different images, with 3 different patterns: strong edge, strong texture, and weak texture, respectively. The red point represents the centre of the studied 8×8 patch.

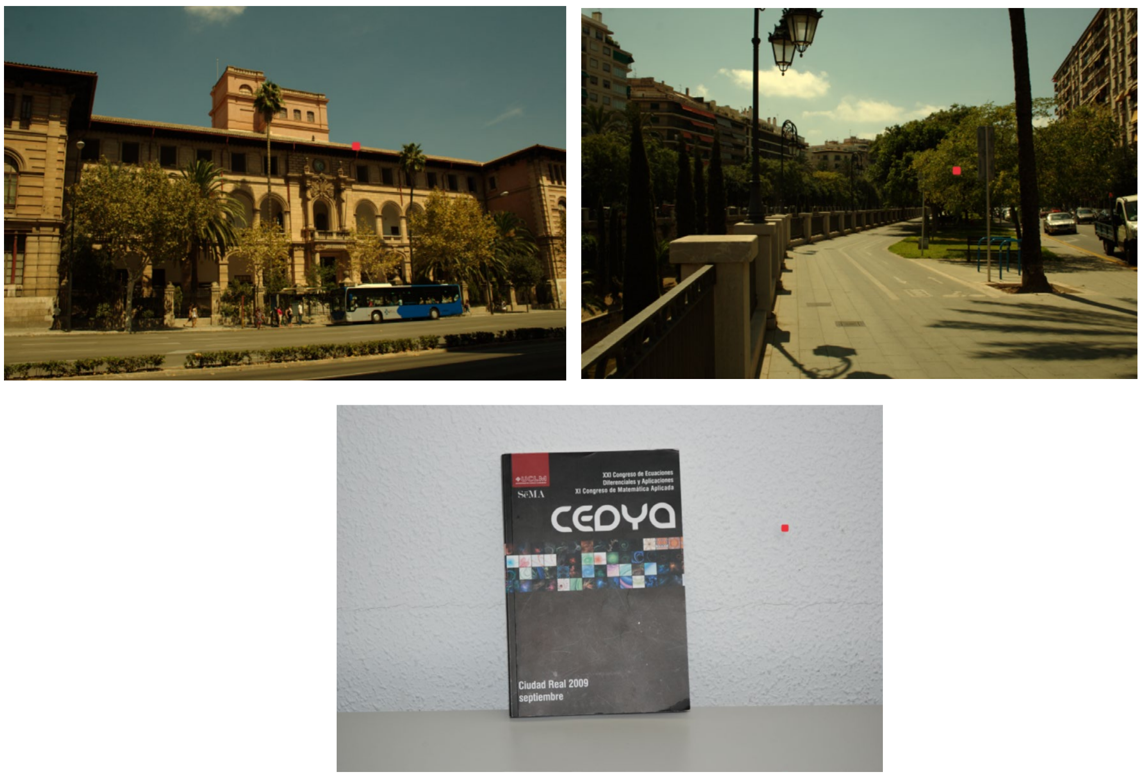

Figure 1: From left to bottom: edge image, textured image, plain image

Self-similarity hypothesis

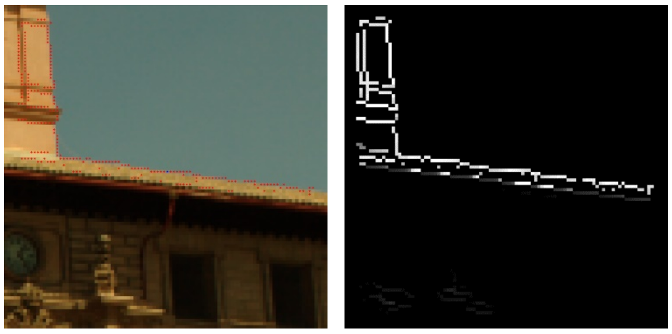
Figure 2: Centres of similar patches and their distributions for edge image

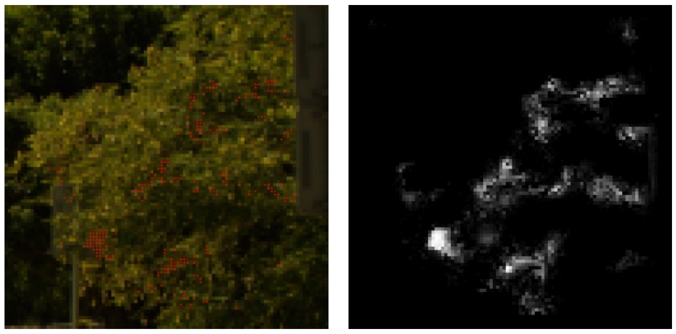
Figure 3: Centres of similar patches and their distribution for the textured image

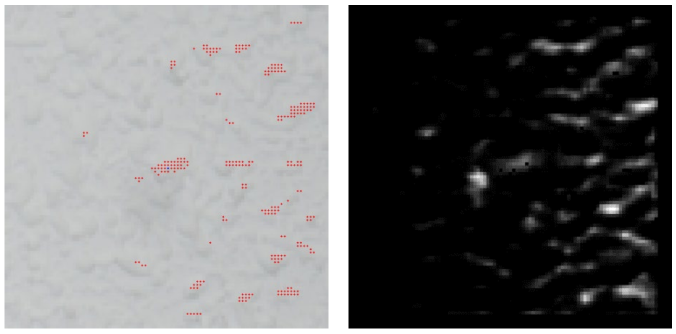
Figure 4: Centres of similar patches and their distribution for the plain image

The first experiment consists in finding similar patches. Those patches were obtained when looking in window 100×100. The distribution helps us to determine what the similar patches represent.

On the edge image (see Figure 2), the similar patches are all located on the edge of the roof and of the building. Nevertheless, we notice that all the patches found do not necessarily lie on a border with the sky, while we have not applied any normalisation (except rotation).

On the foliage image (see Figure 3), we notice that the majority of similar patches are located in the dark part of the tree.

Again, on the plain image (see Figure 4), there are patches similar to the original patch, especially located on paint bumps.

In conclusion, we can see that each patch in each image does indeed contain many similar patches. This similarity is based on several criteria, such as texture, area, colour and contrast of the original patch. Even with low texture, the patches look consistent with each other.We will be able to validate the hypothesis of self-similarity if these patches are indeed very close to the original patch, which is what we will look at.

We will now try to validate the self-similarity hypothesis by looking at similar patches individually. The Figure 5 shows the similar patches in each image. I have taken the rotationally invariant patches.

We can start by noticing that the edge patches are very consistent with the original image. The first 50 patches all show a strong contrast between the sky and a piece of building. Gradually, this contrast diminishes in the image, so that we end up with patches containing only a strong contrast, with a shadow for example.

On the other hand, on the second image, we notice that the similar patches all have the same texture but also a very large variance. The idea of foliage is present everywhere but you never get exactly the same foliage.

Finally, on the second image, it is not surprising to see that all the similar patches are uniform patches with very little variance.

Thus, this study shows us that similar patches can be found in all three images and that the self-similarity hypothesis can be validated. Furthermore, this study also highlights the limitations of Non-Local Means Denoising. We understand that this algorithm works very well on areas with many similar patches (the edge and plain image) but it would have some difficulties on the textured image.

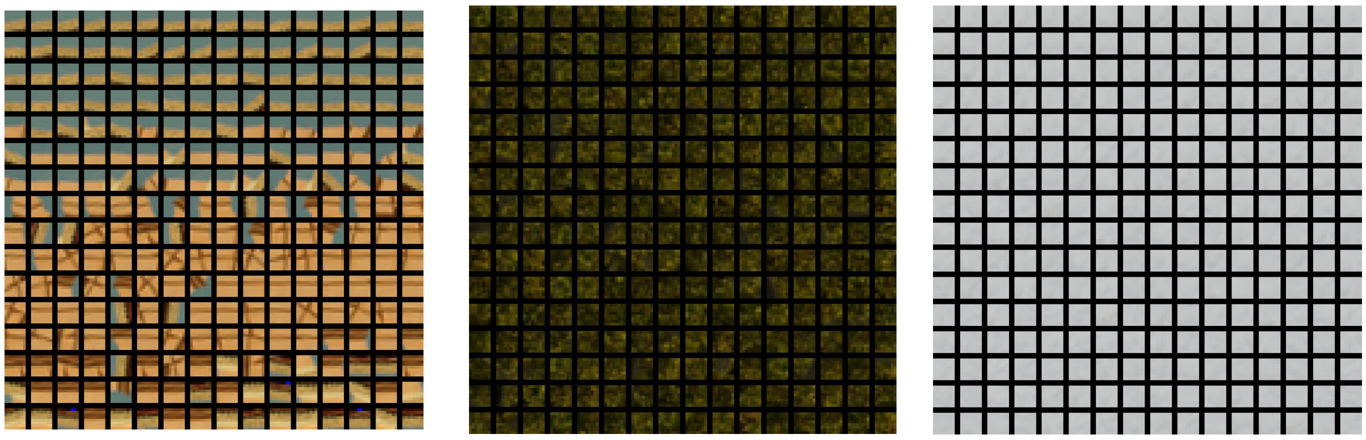

Figure 5: Similar patches per image

Gaussianity hypothesis

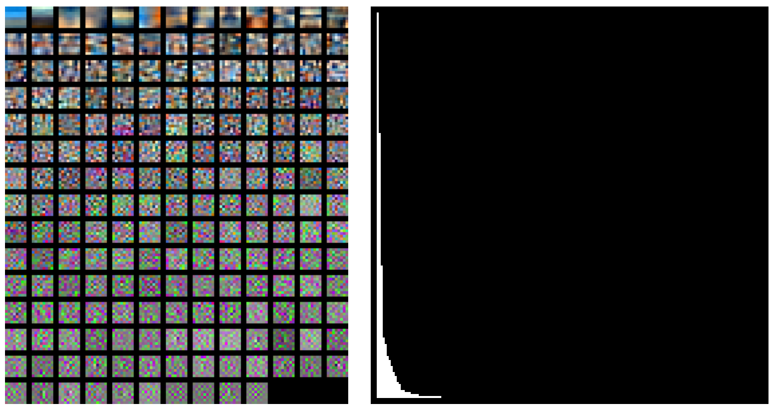

Figure 6: PCA eigenvectors and eigenvalues for edge image

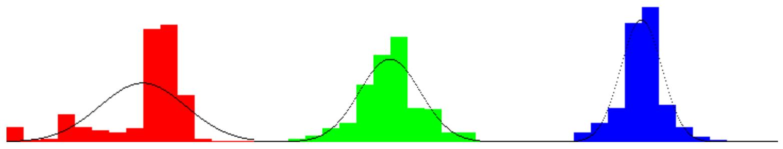

Figure 7: PCA projections. Anderson-Darling normality test (p-values):0.0000, 0.0002, 0.0000

The next objective is to test the Gaussianity hypothesis. To do so, we will see if the patches can easily be represented by a PCA and a Gaussian mixture model (with 2 components).

On the edge image, we notice that the PCA is very sparse (see Figure 6). All the eigenvalues are almost 0 from the 20th value. One has the impression that only the first line of eigenvectors really represents the image.

Moreover, the standard deviation of the PCA and the Gaussian Mixture are totally different (30 vs 25). The Gaussian distributions are not redundant, and therefore cover very different patches. One component covers the edges while another covers the shadows. Moreover, the Anderson-Darling normality test invites us to reject the Gaussianity hypothesis (see Figure 7).

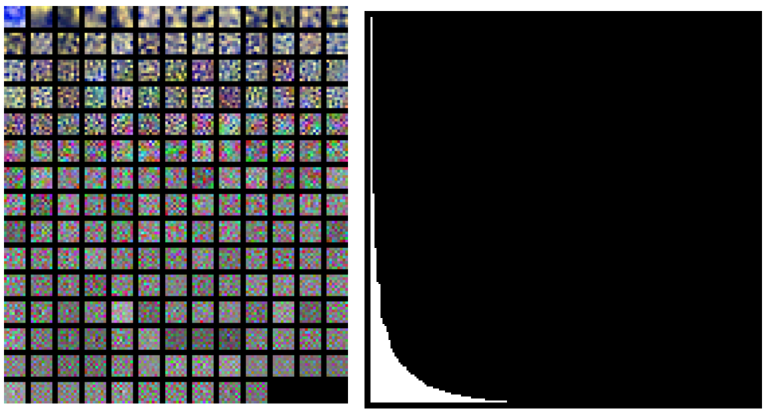
Figure 8: PCA eigenvectors and eigenvalues for the textured image

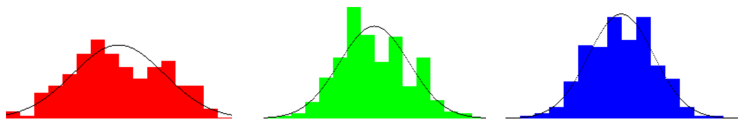
Figure 9: PCA projections. Anderson-Darling normality test (p-values):0.0005, 0.0045, 0.7241

For the textured image, we notice that the PCA is much less sparse (see Figure 8). Indeed, the image is much more complex than a simple contrast. The standard deviation of the PCA and the 2 Gaussian Mixtures are quite close, showing that the 2 methods represent the patches quite well. Moreover, the 2 Gaussian Mixtures sometimes represent the same data, despite a polarisation by one of the two Gaussians.

Finally, the Gaussianity hypothesis is partially rejected, at least on the red and green channels (the majority in the patches).

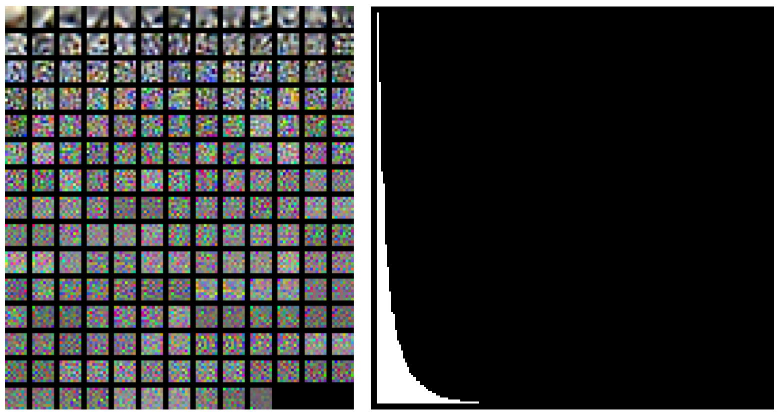
Figure 10: PCA eigenvectors and eigenvalues for the plain image

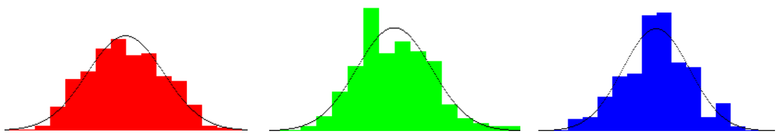
Figure 11: PCA projections. Anderson-Darling normality test (p-values):0.1944, 0.0106, 0.0372

For the plain image, we notice once again that the PCA is much less parse than the edge image (see Figure 10) . On the other hand, the eigenvectors quickly seem to represent only noise. The two components of the Gaussian Mixture Model play exactly the same roles without distinction.

In this image, it is clear that the Gaussianity hypothesis can be retained. Indeed, if we put all these patches side by side, it is as if we had taken a picture of a uniform grey wall without contrast, with only noise. Thus, the noise follows a Poisson distribution, which can be modelled by a Gaussian. Finally, validating the Gaussianity hypothesis in the patches is equivalent to validating the Gaussianity hypothesis of the noise in the image formed by all similar patches.

In the end, by playing with the hyperparameters, we can modify our results.

The method of extracting the points will modify the similar patches found. For example, if we only select Harris Point or Canny Points, we only select patches containing a strong contrast of a certain elongation. This can also change the calculation times as only a part of the points in the image are considered. This acts like a filter.

We can also choose to normalize the mean and/or the variance of the patches. This will tend to favour patches at a distance which may experience different illumination from the original patch.

The NL-Bayes algorithm

Description of NL-Bayes

The NL-Bayes algorithm is an extension of the NL-Means algorithm. Moreover, it is very close to the BM3D algorithm.

The general objective of the NL-Bayes algorithm is to estimate the noise. To do this, it will repeat the same operation twice on the patch to be denoised. Thus, operation 1 will produce a denoised patch which will serve as an input for operation 2, like an oracle. Each of these operations can be broken down as follows:

1) Search for similar patches and concatenation in 3D
2) Estimation of the noise on these patches
3) If the noise is low, the patch is considered as homogeneous. Otherwise, the noise is modelled by a Gaussian distribution, solution of the Bayesian minimal mean square error.
4) Group all similar patches according to an aggregation procedure

The NL-Bayes algorithm achieves very good performance for low noise and on RGB images.

The Bayesian approach can also help to define a new approach which is NL-PCA. The idea of this method is to perform the PCA on the 3D block, instead of computing the Bayesian estimator. Once again, this method is also very similar to the BM3D method, the only change is in the collaborative filtering by changing the linear transformation with the PCA.

Theoretical comparison with BM3D

The idea behind NL-Bayes is very similar to BM3D. Thus, we can see differences and similarities in each of these algorithms:

- The pre-processing step is the same, with the passage to the luminance-chrominance space.
- The similarity is calculated using only luminance channel in the first step. In the second step, the NL-Bayes uses the estimated channels to calculate the distances. In contrast, the BM3D uses the estimated channels and luminance channel.
- The filtering step is quite different. On the one hand, BM3D uses a hard thresholding and a linear transformation for the estimation. On the other hand, NL-Bayes differentiates the case of a homogeneous patch, and applies a Bayesian approach otherwise. Both algorithms have 2 steps with the calculation of an oracle.
- Both algorithms undergo the same aggregation step, although the weights are different.
- The NL-Bayes algorithm benefits from an improvement that prevents it from using already denoised patches.

Thus, the NL-Bayes algorithm presents an improvement in terms of completeness on small patches because its filtering step is solved by a simple matrix inversion.

Visual comparison with BM3D

We will apply NL-Bayes and BM3D to different scenes: an artificial scene with a checkerboard and a natural scene with textures.

For the checkerboard image (see Figure 12), we see that there is a huge variability in the difference images. The denoised image with NL-Bayes has very strong edges, compared to BM3D. This result in higher PSNR for the BM3D image. However, visually speaking, it seems like the result are quite similar, this is why we are going to study different levels of noise in natural images.

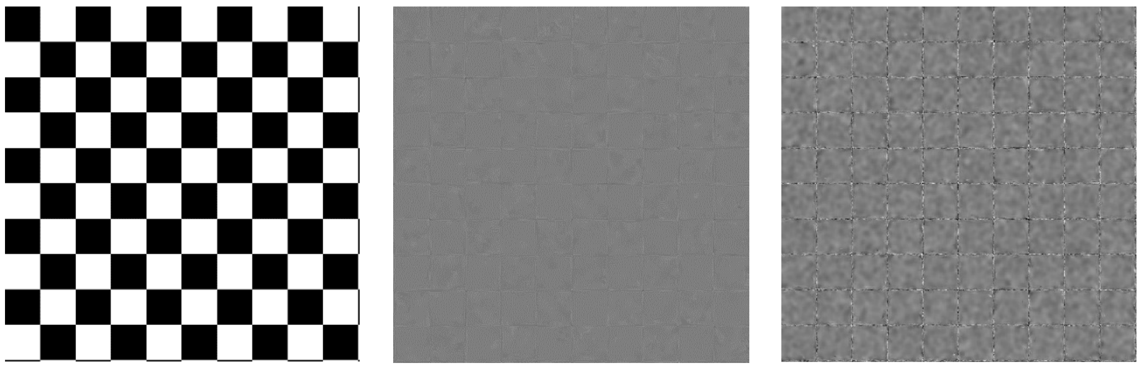
Figure 12: Left: Original image. Middle: Difference for BM3D (PSNR: 39.9963dB). Right: Difference for NL-Bayes (PSNR: 37.148dB)

In this second scene (see Figure 13), we notice that both models produce similar performances (about 29.8dB). On the other hand, the visual results have nothing to do with it, which is greatly emphasized by the difference images.

Thus, we notice that the edges on the NL-Bayes denoised image are much sharper, which is very pleasant to look at (see Figure 15). The roof of the building is clean. On the other hand, for this level of noise, we also notice that this algorithm will produce thick spots, especially on the homogeneous parts. The algorithm handles differently the homogeneous parts, like the sky. Looking at the sky, I feel like I'm watching a badly compressed video stream on Netflix when I'm in the subway. So the result is globally better on the denoised image by BM3D.

For lower noise levels, these spots disappear little by little (see Figure 14). So the NL-Bayes algorithm becomes better at low noise.

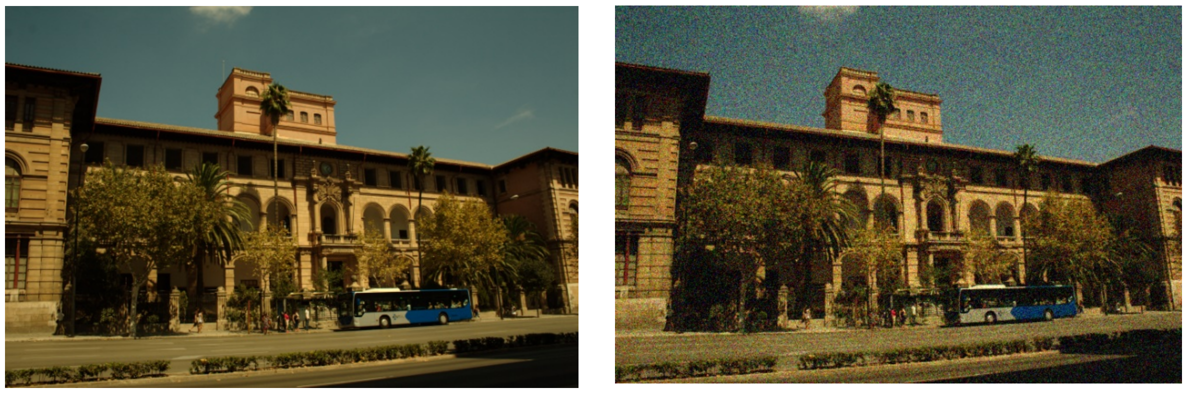
Figure 13: Left: Original image. Right: Noisy image.

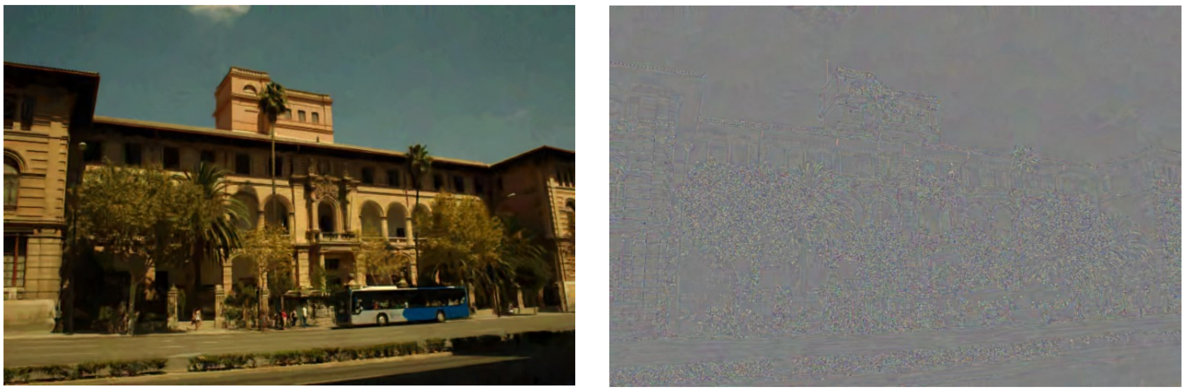
Figure 16: Denoised image and difference with BM3D (PSNR: 29.7421dB, $\sigma = 30$)

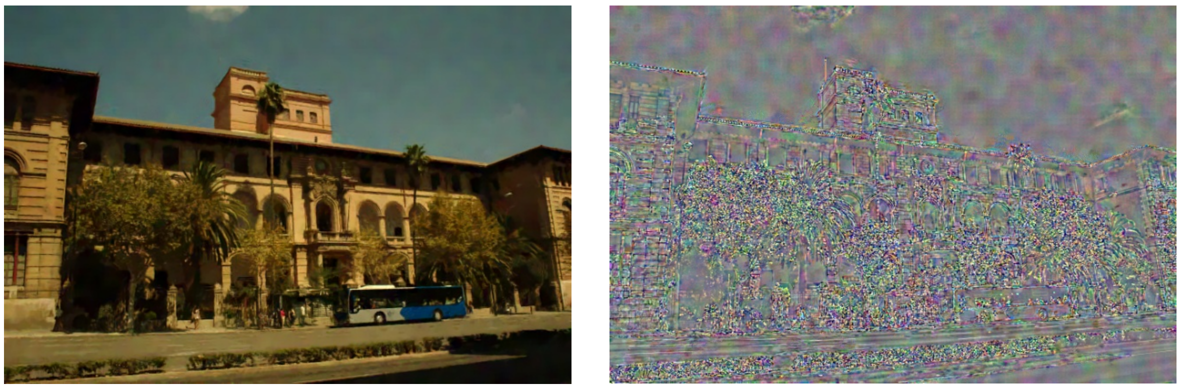
Figure 15: Denoised image and difference with NL-Bayes (PSNR: 29.8417dB, $\sigma = 30$)

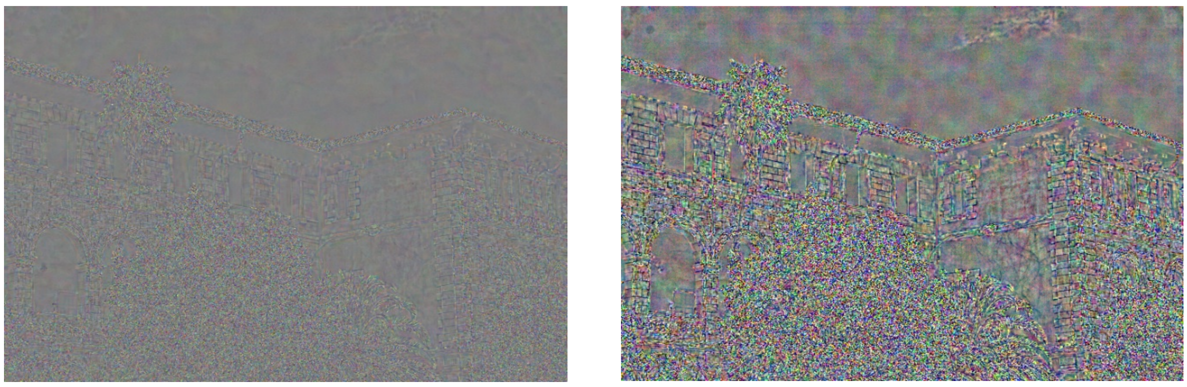
Figure 14: Left: difference for BM3D (PSNR: 35.4762dB, $\sigma = 10$). Right: Difference for NL-Bayes (PSNR: 35.9953dB, $\sigma = 10$)