Video Denoising via Empirical Bayesian Estimation of Space-Time Patches

Pablo Arias, Jean-Michel Morel

Noname manuscript No.
(will be inserted by the editor)

Video denoising via empirical Bayesian estimation of space-time patches

Pablo Arias · Jean-Michel Morel

Received: date / Accepted: date

Abstract In this paper we present a new patch-based empirical Bayesian video denoising algorithm. The method builds a Bayesian model for each group of similar space-time patches. These patches are not motion compensated, and therefore avoid the risk of inaccuracies caused by motion estimation errors. The high dimensionality of spatio-temporal patches together with a limited number of available samples, poses challenges when estimating the statistics needed for an empirical Bayesian method. We therefore assume that groups of similar patches have a low intrinsic dimensionality, leading to a *spiked covariance model*. Based on theoretical results about the estimation of spiked covariance matrices, we propose estimators of the eigenvalues of the *a priori* covariance in high dimensional spaces as simple corrections of the eigenvalues of the sample covariance matrix. We demonstrate empirically that these estimators lead to better empirical Wiener filters. A comparison on classic benchmark videos demonstrates improved visual quality and an increased PSNR with respect to state-of-the-art video denoising methods.

1 Introduction

Advances in video sensor hardware have steadily improved the acquisition quality. However, due to the reduction in the price of cameras and data storage, video cameras are being used increasingly often in unfavorable conditions, such as low lighting. This results in high levels of noise, which negatively affects the visual quality of the video and hinders its use for any application.

This work is partly founded by BPIFrance and Rgion Ile de France, in the framework of the FUI 18 Plein Phare project; by the Office of Naval research by grant N00014-17-1-2552; by ANR-DGA project ANR-12-ASTR-0035; and by ANR-DGA project ANR-14-CE27-001 (MIRIAM).

Pablo Arias
Centre de mathématiques et de leurs applications (CMLA),
ENS Cachan, Université de Paris-Saclay,
61 Av. du Président Wilson,
F-94235 Cachan, France
E-mail: pablo.arias@cmla.ens-cachan.fr

Jean-Michel Morel
Centre de mathématiques et de leurs applications (CMLA),
ENS Cachan, Université de Paris-Saclay,
61 Av. du Président Wilson,
F-94235 Cachan, France
E-mail: jean-michel.morel@cmla.ens-cachan.fr

The denoising of videos has received less attention than still image denoising in the literature. In principle, denoising videos should be easier than still images, since the strong temporal redundancy along motion trajectories favours the denoising task. However, this additional form of redundancy creates challenges. On the one hand, the amount of data is much larger than for still images, and efficient ways of navigating through this data are needed. On the other hand, the output of the denoising algorithm should be temporally consistent with the unknown motion of the original sequence, which is an essential quality criterion [37, 55].

Early works proposed temporal or spatio-temporal filters which either compensate for motion using estimated motion trajectories or used some kind of mechanism to adapt to the temporal changes in the video signal at a fixed location. We refer the reader to [5] and references therein for more details. Other video denoising works apply temporal filtering to the wavelet coefficients of the frames [27,62]. Some methods do not distinguish between the temporal and spatial dimensions, and treat the video as a volume, which is denoised in a transformed domain [51, 54].

Until the advent of patch-based approaches, methods without motion compensation failed in the presence of moderate motion, and their main interest resided in their simplicity and low computational cost. The first state-of-the-art results obtained without motion estimation were reported in [6], with the non-local means algorithm. The authors argued that for their method, based on averaging similar patches, motion estimation was not only unnecessary but counterproductive: while self-similar regions are problematic for motion estimation, they are a source of a large number of similar patches across different frames. Even if the motion trajectory could be reliably estimated, it makes more sense to use all similar patches, rather than using only the ones on the trajectory. A similar approach was followed by the authors of [14]. This method is based on collaborative filtering of similar patches in a spatio-temporal neighborhood. The methods in [6] and [14] are extensions to video of still-image patch-based denoising algorithms (non-local means [7] and BM3D [15] respectively). They are based on filtering similar 2D patches searched for in a 3D spatio-temporal neighborhood. These methods exploit both spatial and temporal redundancy. However, since each 2D patch is processed independently, there is no mechanism to impose coherence along trajectories. This results in flickering artifacts which become noticeable for high levels of noise.

An interesting alternative was proposed in [37], based on non-local means. To denoise a target 2D patch, its forward and backward trajectories are estimated using an optical flow algorithm. The patch is denoised as a weighted average of the patches along the trajectory and their nearest neighbors. The weight considers the patch similarity like in the non-local means method, multiplied by a coefficient that decreases smoothly with the temporal distance to the target patch. In this way, 2D patches in the same trajectory are denoised using a smoothly varying non-local average of similar 2D patches, resulting in a video with a high temporal consistency.

A natural next step is to consider 3D spatio-temporal patches. This has two main advantages. First it provides a mechanism to impose some temporal consistency on the result, since then, the temporally contiguous 2D temporal slices of a spatio-temporal patch are filtered jointly. Second, a 3D patch has a higher number of pixels resulting in a reduction of the noise in the distance between patches. To get the same reduction in the distance noise with 2D patches one needs to increase their spatial size which typically leads to a smaller number of similar patches. Two types of 3D patches have been used in the literature: rectangular and motion-compensated. Motion-compensated 3D patches follow a motion trajectory previously estimated.

Motion-compensated patches have the benefit that their temporal slices are highly correlated. As a consequence, temporal filtering can be easily performed by filtering inside the patch in the temporal direction. Because of this, methods that use motion-compensated 3D patches only look for similar motion-compensated 3D patches starting in the same frame.

In [38, 39] the authors introduced V-BM4D, an extension of BM3D to video by collaborative filtering of similar motion compensated 3D patches. Trajectories are computed with a block matching strategy based on the sum of squared differences (SSD) and a temporal regularization term, which favours trajectories with small velocity and low acceleration. Recently, Buades et al. [8] proposed to denoise each frame by warping the neighboring frames to match the target frame via an optical flow method. This defines a 3D volume with almost identical frames (in the ideal case). A patch-based denoising method is then used to denoise the central frame in the warped volume. Patch similarity is determined using 3D patches in the warped volume, which is equivalent to consider motion compensated patches. The denoising of each 2D patch in the target frame results from a PCA model learnt from the 2D frames of the similar 3D patches.

The main difficulty in using motion-compensated patches is obtaining an accurate motion estimation in the presence of noise. An alternative which does not require motion estimation is to consider rectangular 3D patches. Each rectangular 3D patch results from the combination of a 2D texture and a motion pattern. For arbitrary motion it would seem that each combination of texture and motion is unlikely to repeat across the video. Thus rectangular 3D patches should have less repeatability in the video compared to motion compensated patches. This objection is mitigated in two ways. First, if the motion is uniform, rectangular 3D patches will be similar along the trajectory. Second, if the motion pattern is spatially homogeneous in a region of the image, then the image self-similarity still extends to space-time patches starting at the same frame.

Indeed, many authors have considered rectangular 3D patches. An improved version of non-local means, which adaptively controls the shape and size of the spatio-temporal search region is proposed in [3]. The authors compare the performance obtained with 2D and 3D patches. Sutour et al. [56] proposed to add an adaptive spatial regularization to non-local means, and to apply it to images and videos, also with 3D patches. In [49, 50] the authors extend the K-SVD [20] image denoising method to video by learning a dictionary of spatio-temporal rectangular 3D patches. A multi-scale version was proposed in [42]. Ghoniem et al. [23] model non-local interactions in a video as a weighted graph where nodes are associated to 3D patches and the weights on the edges are given by patch similarity. A graph regularization framework is then proposed and applied to several inverse problems. In [41] a version of V-BM4D, called BM4D, using rectangular 3D patches is applied to the restoration of volumetric images. The authors report results on video denoising as well, with a performance inferior to that of V-BM4D, particularly in sequences with motion. A similar approach, using shape adaptive patches was proposed in [36] for grayscale sequences.

In this work we present a new empirical Bayesian patch-based video denoising method using 3D rectangular patches. The proposed method assumes that similar patches can be modelled as independent, identically distributed (IID) samples from an unknown *a priori* distribution. Each patch is denoised by computing the linear minimum mean square error estimator (LMMSE) with the estimated first and second order moments of the prior. Equivalently, the proposed approach can be described as a *maximum a posteriori* (MAP) under the assumption that the prior distribution is Gaussian.[1]

In recent years several works have proposed Gaussian models (or Gaussian mixture models) as priors for image patches, achieving state-of-the-art results in image denoising and other inverse problems. However, the potential of these techniques for video restoration remains mostly unexplored.

[1] We refer to our method as an empirical Bayesian approach because although it is based on a Bayesian model for patches, the parameters of the prior are determined from the data using *frequentist estimators*. The empirical (or data-driven) Bayesian approach is a usual choice when the parameters of the prior are unknown, but it is a departure from a strict Bayesian methodology which would require a hyperprior to estimate the parameters (see [1] for a closely related method considering a Bayesian estimation of the parameters of the prior).

A Gaussian mixture model was proposed in [63] as a fixed prior for image patches. The mixture comprising hundreds of Gaussians, is learnt off-line from a large patch database. In [11, 59] an image dependent mixture is learnt for each image. These works consider a single GMM for the whole image. Extending these approaches to video is challenging because of the high variability of spatio-temporal patches. Thus we shall stick to the idea of using video redundancy to build a local Gaussian model for each patch from its most similar patches as in [1, 32, 61]. Such an approach is applied in [8] to video denoising, but restricted to 2D patches. The authors argue that the main limitation for considering 3D patches is that, as the dimensionality of the patch grows, the number of similar patches required to estimate the covariance matrix grows larger than the number of similar patches available.

However, we can expect that similar patches have only a few modes of variation, resulting in a low intrinsic dimensionality, or equivalently a low-rank covariance matrix. Under this assumption we propose estimators of a low-rank *a priori* covariance matrix in high dimensional spaces, which are motivated by recent theoretical results and supported by an empirical analysis. We apply these estimators to derive empirical Wiener filters adapted to sets of similar 3D patches. We tested with 3D patches of dimensionality $d = 100$ and $d = 200$ and different combinations of spatial and temporal sizes. The proposed approach produces results which compare favourably to the current state-of-the-art, both quantitatively and qualitatively, specifically in terms of temporal consistency.

A preliminary version of this work using 2D patches was presented in [2]. Here we generalize it by considering 3D patches together with an improved estimation of the eigenvalues of the low-rank covariance matrices.

The rest of the paper is organized as follows: in §2 we describe the proposed algorithm. In §3 we address the problem of estimating a low-rank covariance matrix in high dimensional spaces and propose two simple estimators derived from the asymptotic results of [18]. In §4 we explain the selection of the parameters. Some results, including a comparison with the state-of-the-art methods, are shown in §5. Concluding remarks are given in §6.

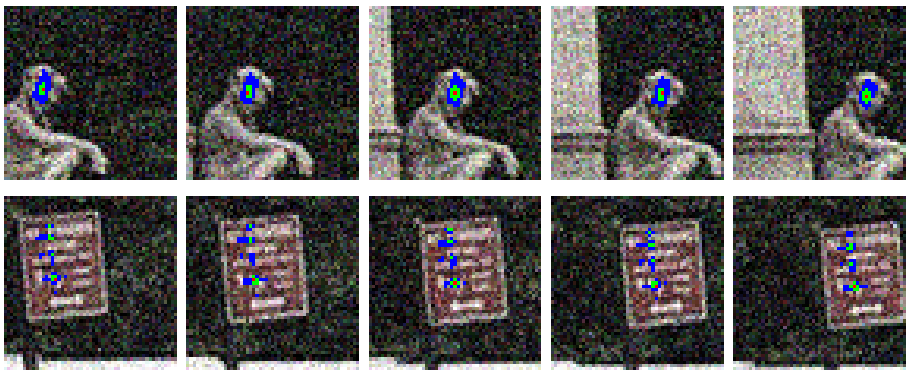

Fig. 1: Examples of sets of 200 nearest neighbors of two reference spatio-temporal patches, of size $9 \times 9 \times 4$. The images in each row show the five frames of a $37 \times 37 \times 5$ search region. In red we show the positions of the 5 nearest neighbors (one of them, in the center of the middle frame is the reference patch), in green the next 40 and in blue the rest. The points shown correspond to the top-left pixel of the first spatial slice of each patch. To highlight the position of the patches, the color of the images has been attenuated.

2 Bayesian video denoising

Our main assumption, following [32], is that video patches similar to a given patch are independent random samples drawn from some *a priori* distribution. The problem of denoising each patch can then be formulated as a problem of optimal Bayesian inference, where the parameters of the prior are learnt from the noisy data.

2.1 A non-local Bayesian principle

Consider a grayscale video $u : \Omega \times \{1, \cdots, T\} \to \mathbb{R}$, where Ω is the spatial domain (a rectangular discrete grid). We assume that v is a noisy version of u, contaminated with additive, Gaussian, white noise (AGWN) z of known variance σ^2,

$$v = u + z.$$

We denote a rectangular 3D patch of size $s_x \times s_x \times s_t$ of the noisy video v as $\boldsymbol{q}$, and the corresponding patch in the clean video u as $\boldsymbol{p}$. We consider the patches as vectors in $\mathbb{R}^d$, with $d = s_x^2 s_t$. We have $\boldsymbol{q} = \boldsymbol{p} + \boldsymbol{z}$, where $\boldsymbol{z} \sim \mathcal{N}(0, \sigma^2 I_d)$ is a patch of AGWN (I_d denotes the $d \times d$ identity matrix).

For a noisy patch $\boldsymbol{q}$, we select the n closest 3D patches $\boldsymbol{q}_1 = \boldsymbol{q}, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_n$ according to the Euclidean distance in $\mathbb{R}^d$. The search is conducted in a spatio-temporal region centered at $\boldsymbol{q}$. The spatial section of the search region is of size $w_x \times w_x$ and it extends for w_t frames. The main assumption justifying a Bayesian method is that the corresponding patches in the clean video, $\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_n$, are independent random samples from an *a priori* distribution with mean $\mathbb{E}\{\boldsymbol{p}\} = \boldsymbol{\mu}$ and covariance $\mathbb{E}\{(\boldsymbol{p} - \boldsymbol{\mu})(\boldsymbol{p} - \boldsymbol{\mu})^T\} = C$.

We can then estimate the clean patches $\boldsymbol{p}_i$ using the linear estimator minimizing the expected mean square error (MSE), or LMMSE:

$$\widehat{\boldsymbol{p}}_i = \boldsymbol{\mu} + C(C + \sigma^2 I_d)^{-1}(\boldsymbol{q}_i - \boldsymbol{\mu}) \tag{1}$$

If we assume that the *a priori* distribution is Gaussian, then this estimator minimizes the MSE across all estimators, and coincides also with the *maximum-a-posteriori* (MAP) estimator.

The matrix $C(C + \sigma^2 I_d)^{-1}$ in the above equation is the known Wiener filter and can be diagonalized in the basis of principal directions of the covariance matrix C. Let $C = U \Lambda U^T$ denote the spectral decomposition of C, where U is the orthogonal matrix of eigenvectors and $\Lambda = \mathrm{Diag}(\lambda_1, \lambda_2, \ldots, \lambda_d)$ is a diagonal matrix with the eigenvalues sorted in non-increasing order. The eigenvalue λ_i is the variance of the *a priori* distribution along the ith principal direction. Then we have that

$$U^T(\widehat{\boldsymbol{p}} - \boldsymbol{\mu}) \quad = \quad \Lambda(\Lambda + \sigma^2 I_d)^{-1} \;\; U^T(\boldsymbol{q} - \boldsymbol{\mu}).$$

The diagonal operator $S = \Lambda(\Lambda + \sigma^2 I_d)^{-1}$ has the coefficients

$$s_{ii} = \frac{\lambda_i}{\lambda_i + \sigma^2} = (1 + \mathrm{snr}_i^{-1})^{-1} \tag{2}$$

which apply a linear shrinkage of the principal components depending on the signal-to-noise ratio $\mathrm{snr}_i = \lambda_i / \sigma^2$ of each component.

The LMMSE estimator requires the knowledge of the mean and covariance of the *a priori* distribution, which have to be estimated from the data. The resulting data-dependent filter is sometimes referred to as *empirical Wiener filter*.

The mean and covariance from the *a posteriori* distribution can be estimated from the data. Since the noise has zero mean and is independent from u, we have that

$$\mathbb{E}\{\boldsymbol{q}\} = \mathbb{E}\{\boldsymbol{p}\} = \boldsymbol{\mu}, \qquad \mathbb{E}\{(\boldsymbol{q}-\boldsymbol{\mu})(\boldsymbol{q}-\boldsymbol{\mu})^T\} = C_{\boldsymbol{q}} = C + \sigma^2 I_d.$$

These can be estimated as the sample mean and sample covariance matrix

$$\widehat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{q}_i \quad \text{and} \quad \widehat{C}_{\boldsymbol{q}} = \frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{q}_i - \widehat{\boldsymbol{\mu}})(\boldsymbol{q}_i - \widehat{\boldsymbol{\mu}})^T. \tag{3}$$

These correspond to the *maximum likelihood* estimators (MLE) if the *a priori* is Gaussian (in which case the *a posteriori* distribution is Gaussian as well, since the noise is also Gaussian).[2]

The above method is used in [32] for still image denoising, where the *a priori* covariance matrix is estimated as $\widehat{C} = \widehat{C}_{\boldsymbol{q}} - \sigma^2 I_d$. This may lead to a matrix which is not positive semi-definite (and therefore not a valid covariance matrix). We will address this issue later in Section 3 where we propose a more principled estimator, valid in cases in which the space dimensionality is high.

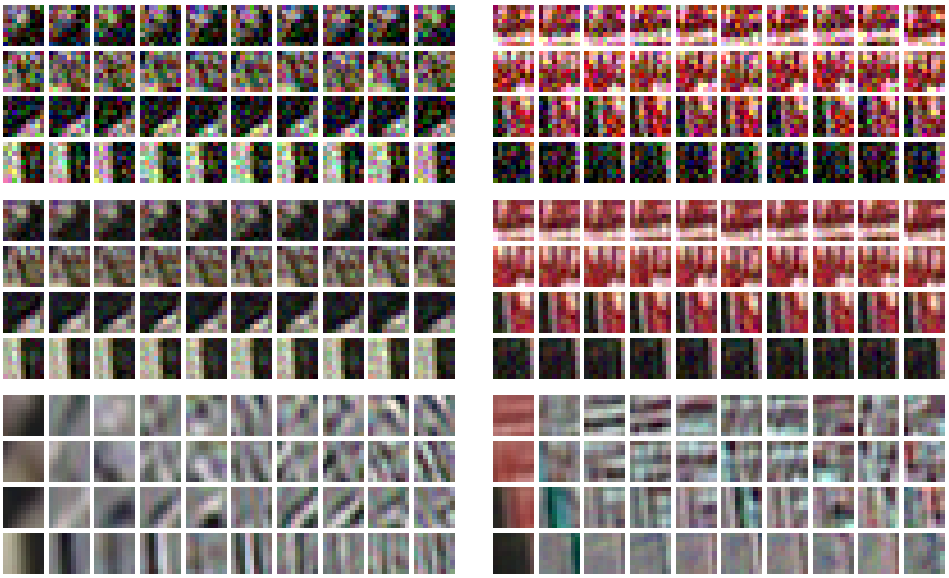

Fig. 2: For the two reference patches displayed in Fig. 1 we show the 10 nearest neighbors (top four rows), the corresponding MAP estimates (middle four rows) and the sample mean and first 9 principal directions (bottom four rows). The four rows in each case correspond to the four frames of the $9 \times 9 \times 4$ patches displayed vertically.

Figures 1 and 2 show two examples of sets of similar $9 \times 9 \times 4$ patches. Figure 1 displays the positions of the similar 200 patches found in a local search region and Figure 2 shows the 10 nearest neighbors in more detail. Each patch is displayed by vertically stacking its four 9×9 temporal slices. For the two examples the figure shows the noisy patches, the result of the LMMSE estimation

[2] In [32] the *unbiased* estimator for the covariance matrix is used, which differs from the MLE in that the sum is divided by $n-1$ instead of n. In practice, except for small values of n, the choice between these two estimators has little effect on the results. Here we prefer to use the MLE because it is the one studied by Paul in [18]. Later, in §3.2 we will exploit Paul's results to derive estimators for the eigenvalues of the covariance matrix.

and the mean patch together with the leading 9 principal directions. The Gaussian models were learnt from the $n = 200$ similar patches shown in Figure 1. The temporal slices of the principal directions show oscillatory patterns, with increasing frequency, oriented in the direction of the dominant edges in the patches. In some cases these patterns are localized in a smaller region. Note that even if the temporal slices of the patch may seem unrelated (particularly for the example on the left), the model captures the correlations between them, and in this sense it differs from a decoupled model in which the frames are independent. Taking into consideration these temporal correlations is crucial for the temporal consistency of the result.

2.2 Description of the algorithm

The pseudo-code of the proposed video denoising method is described in Algorithms 1 and 2. As in [14,15,32,39] we perform two stages. The first stage computes a *basic* estimate $\widehat{u}^{(1)}$. The second stage computes the final estimate $\widehat{u}^{(2)}$ using the basic estimate as an oracle: patches from the basic estimate are used to compute the patch distances and to estimate the mean and covariance matrix of the *a priori* distribution. For the latter, we consider the residue of the basic estimate as white Gaussian noise with variance $\sigma^{(1)}$. This of course is not true, but it is a rough approximation useful to set the parameters of the 2nd step. A similar approximation is done in [63] also to control a parameter in an iterative denoising method.

Note that all the patches of each group are denoised, and these estimates are aggregated via averaging to form the denoised image. There are two aggregation levels. On the one hand, the value of a pixel in the denoised image is computed as the average of all denoised patches that overlap the pixel. In addition, if a patch belongs to several groups, it will be denoised multiple times and these estimates are averaged as well. The aggregation is a key element of patch-based denoising methods [13, 34].

Algorithm 1 Video NL-Bayes

Require: Noisy video v, noise σ

Ensure: Final estimate of noiseless video $\widehat{u}^{(2)}$

1: $\widehat{u}^{(0)} = v$, $\sigma^{(0)} = \sigma$

2: $\widehat{u}^{(1)} = \texttt{video\_nlbayes\_step}(v, \sigma, \widehat{u}^{(0)}, \sigma^{(0)})$

3: $\sigma^{(1)} = \texttt{estimate\_noise}(\widehat{u}^{(1)})$

4: $\widehat{u}^{(2)} = \texttt{video\_nlbayes\_step}(v, \sigma, \widehat{u}^{(1)}, \sigma^{(1)})$

2.3 Implementation details

Handling of color. We express the video in a luminance-chrominance color space, by applying the opponent color transform (see [15]). In the first step, patch distances are computed using the luminance only, in the second step the three channels are used. Using the n most similar patches, a group is built for each channel. These groups are filtered independently (a Gaussian model is learnt for each of them).

This differs from [32], where for the second step a joint *a priori* Gaussian model is used for the color patches. In principle this should allow for a better treatment of color by capturing the

Algorithm 2 Video NL-Bayes step

Require: Noisy video v, noise σ, basic estimate $\widehat{u}^{(1)}$, basic noise $\sigma^{(1)}$

Ensure: Final estimate of noiseless video $\widehat{u}^{(2)}$

Set $\mathcal{P} = \{\boldsymbol{q} : \boldsymbol{q} \text{ patch of } v\}$ // eligible reference patches
while $\mathcal{P} \neq \emptyset$ **do**
  Get a patch $\boldsymbol{q}$ from $\mathcal{P}$ // reference patch
  Retrieve the n nearest neighbors to $\boldsymbol{q}$ in a spatio-temporal volume around $\boldsymbol{q}$. The distance is computed between the basic estimates $\widehat{\boldsymbol{p}}^{(1)}$.
  Estimate $\widehat{\boldsymbol{\mu}}^{(2)}$ and $\widehat{C}^{(2)}$ from the basic estimates $\widehat{\boldsymbol{p}}_i^{(1)}$ (see §3).
  for all n neighbors $\boldsymbol{q}_i$ of $\boldsymbol{q}$ **do**
    Compute LMMSE estimate $\widehat{\boldsymbol{p}}_i^{(2)}$, Eq. (1).
    Aggregate $\widetilde{\boldsymbol{p}}_i^{(2)}$ on $\widehat{u}^{(2)}$.
    Remove $\boldsymbol{q}_i$ from $\mathcal{P}$. // for speed-up
  end for
end while

correlations between the color channels. We found that this option leads to slightly better results (an improvement of 0.2dB up to 0.4dB), but at a much higher computational cost. This is due to the fact that if the channels are treated independently the patch dimensionality is $s_x^2 s_t$, whereas a joint treatment requires a single Gaussian model of tripled dimensionality. As we show in the next section, the processing of a group of patches scales with the patch dimensionality d as $\mathcal{O}(d^2)$.

Patch distance threshold. When building the group of similar patches during the second stage, more than n patches are allowed if their distances with respect to the reference patch are smaller than a threshold ρ. As in [33], we set $\rho = 4$. The distance between patches is normalized by the number of elements in the patch, the number of channels, and the values of each channel are in the range $[0, 255]$. [3] Thus ρ can be interpreted as the maximum average pixel difference between two similar patches. It is a small value to guarantee that only very similar patches are considered.

In the same way the distance threshold ρ is used to include a larger number of similar patches, some methods have mechanisms to exclude patches if they are too far away from the reference patch. In [46] a Gaussian is estimated using a fixed number of nearest neighbors. Then a weighted aggregation is performed which gives small weights to the patches that are not likely according to the learnt model. In this work we did not explore such options.

Sample mean in the second step. In the second step if the sample mean $\widehat{\boldsymbol{\mu}}^{(2)}$ is computed using the patches from the basic estimate, the result tends to be too smooth. Textures and details are better preserved if the noisy patches are used instead. In doing so, however, some noise is kept in flat areas, particularly if the number of similar patches n is small and the noise is high. To balance these two phenomena we perform a simple to determine if a region is flat. Specifically, we compute the sample variance of all pixels of all the patches in the group (we use for this the noisy patches).

[3] The distance threshold ρ is applied in the second step, where the distance is computed using patches from the basic estimate. For this reason (as in [30]) we do not consider a noise dependent threshold.

If this variance is lower than σ^2, then the area is considered flat. This test is essentially a χ^2 test for the variance. For flat patches we compute the sample mean using the basic estimate, whereas for non-flat ones we use the noisy patches. In [32] flat patches are also given a special treatment, and are estimated as the average of all the pixel values of all patches in the set.

2.4 Computational complexity and speed-ups

Processing each group of patches requires (1) searching for the nearest neighbors, (2) estimating the *a priori* covariance matrix and (3) computing the LMMSE estimates. The search for the nearest neighbors is of asymptotic order $\mathcal{O}(w_x^2 w_t d)$. To estimate the covariance matrix and the LMMSE estimates it is convenient to work on the basis of principal directions. As we will see in Section 3 it is usually enough to consider only the $r < d$ leading principal directions. This requires nd^2 operations to compute the sample covariance matrix, rd^2 to compute its r leading eigenvectors and eigenvalues, and nrd operations to apply the Wiener filter on the r principal components. Therefore the computational complexity of processing each group of similar patches is of asymptotic order $\mathcal{O}(w_x^2 w_t d + nd^2 + rd^2 + nrd)$.

The most expensive operations are the computation of the covariance matrix and of its principal directions. There are methods for approximating the r principal directions of a set of points that do not require computing the covariance matrix [26]. Instead these methods efficiently approximate a truncated SVD of the data matrix, and would allow to improve the computational complexity to $\mathcal{O}(w_x^2 w_t d + drn)$.

The computation can be substantially accelerated by reducing the number of groups of patches that are jointly processed. Each group of patches is associated to the reference patch from which the nearest neighbors are computed. We apply two tricks considered in [15, 33] in order to reduce the number of reference patches. The first one consists in visiting only those patches on a coarser spatial grid with vertical and horizontal separation of $s_x/2$. This reduces the number of reference patches by a factor of $s_x^2/4$. The second acceleration trick is to give up taking as reference patch the patches that have been already once filtered in another group (step 9 in Algorithm 2). The actual number of aggregated estimates for each pixel remains anyway high.

Due to these speed-ups, the actual number of groups of similar patches processed depends on the patch size s_x, s_t and n. In practice, for the found optimal parameters (see §4 it ranges roughly from 1% to 5% of the total number of pixels in the sequence.

The running time depends on the patch size. The following table shows the running times of our non-optimized C++ implementation for different patch sizes. The patch size is indicated as $s_x^2 \times s_t$ and the time is expressed in seconds per 10^6 pixels (s/MPx):

patch size	$3^2 \times 2$	$4^2 \times 3$	$7^2 \times 2$	$10^2 \times 1$	V-BM4D-mp
running time	18	65	450	740	350.

These times were computed by averaging the running times obtained for 5 color test sequences for several noise levels. For comparison we also show the running time of V-BM4D [39] using the implementation available online [40] with the "modified profile", which is the one with the highest complexity.

2.5 Related works

Using a Gaussian patch model for image denoising has several antecedents. The BLS-GSM method [48] (Bayes Least Square estimate of Gaussian Scale Mixture) models noiseless "wavelet coefficient neighborhoods" with a Gaussian scale mixture defined as a random scaling of a zero-mean Gaussian

density. The wavelet coefficient neighborhood turns out to be a patch of an oriented channel of the image at a given scale. In [59], the authors introduce a framework for solving inverse problems, based on the assumption that the patches of an image are distributed according to a Gaussian Mixture Model (GMM) which is learnt from the image with an EM-like algorithm. A related approach is proposed in [11], where the patches of the image are clustered according to their "structural similarity" and a Gaussian model is fit to each cluster. A GMM is also used in [63], but it is trained from a database of $2 \cdot 10^6$ patches randomly sampled from the Berkeley database. A hierarchical covariance data structure termed the *covariance tree* was introduced in [25]. The structure captures the covariance of image patches at several scales. The covariance tree can be learnt from a noisy image in the case of denoising, or from a database of images, in which case it can be applied to other inverse problems.

Instead of learning a GMM (or covariance tree) for the whole image in [32] a Gaussian model is fit to each patch and its nearest neighbors. Aguerrebere et al. [1] extended this approach to an hyper-Bayesian model for similar image patches by considering priors on the mean and the covariance matrix. This allows handling more complex inverse problems, where the mean and covariance matrix cannot be directly estimated from the observed data.

Other related approaches are those that use PCA on groups of patches. In [44] a PCA model is learnt for the patches contained on a window sliding over the image. A version of the BM3D algorithm using shape adaptive patches and PCA as a transformed domain was proposed in [16]. In [61] an algorithm that applies PCA on groups of similar patches is introduced. PCA models of patches are is also studied in [19], where the authors compare the performance of different estimators (linear and non-linear) in the domain of the principal components.

Although most of these works focus on white Gaussian noise, PCA models have also been extented to Poisson noise in [52]. This requires a specific PCA model suitable for that type of noise. A multiscale version of [32] was proposed in [35] to handle structured noise, focusing on JPEG compression artifacts. The authors in [31] propose an extension of [32] for the denoising of images taking values on Riemannian manifolds.

Gaussian/PCA models for patches lead to several successful image restoration methods. Yet, their application to video denoising has been limited. In [36], the PCA model of [16] was adapted to video denoising by considering 3D shape-adaptive patches. Buades et al. [8] denoise 2D patches by using a PCA model (in a first step) or a Gaussian prior (in a second step) learnt from similar patches. The similar patches are taken as the 2D slices of 3D motion compensated patches.

All these works assume that the patch point density function can be locally approximated with Gaussians (locally in the patch space). Related ideas of local Gaussian approximations can also be found applied to manifold learning, where a manifold of dimensionality r is approximated as a mixture of Gaussians of rank r [29, 57]. Their success does not necessarily mean that similar patches (or nearby points) do follow Gaussian distributions. Nevertheless, regardless of the exact shape of the distribution of a group of similar patches, the best *linear* estimator in the MSE sense is still the Wiener filter.

This being said, other models for local patches have been proposed recently. For example the authors in [24] propose to penalize a weighted nuclear norm of the matrices formed by stacking a fixed number of similar patches. The resulting estimator is similar to the Wiener filter, in the sense that it is a shrinkage operator on the principal directions.

3 Empirical Wiener filters in high dimensions

Wiener filters have been used extensively for image and video restoration, and denoising in particular. Here we consider the case of a random vector $\boldsymbol{x} \in \mathbb{R}^d$ contaminated with additive white

Gaussian noise $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{z}$ with $\boldsymbol{z} \sim N(\boldsymbol{0}, \sigma^2 I_d)$. The signal of interest $\boldsymbol{x}$ can be a patch of an image or the image itself. In practice, the mean $\boldsymbol{\mu}$ and covariance matrix $C_{\boldsymbol{x}}$ of $\boldsymbol{x}$ are unknown and have to be estimated as well. Suppose a set of n noisy observations $\boldsymbol{y}_1, \dots, \boldsymbol{y}_n$ is available.

In this section we discuss principled ways to estimate the eivengalues of the covariance matrix $C_{\boldsymbol{x}}$, based on results by Paul [18] about the asymptotic behaviour of the spectral decomposition of the sample covariance matrix when both d and n tend to infinity at the same rate, for Gaussian distributions with *spiked covariances*. These results fit well the video denoising setting due to the high dimensionality d of spatio-temporal patches.

The resulting asymptotic estimator defines a threshold below which the *a priori* eigenvalue and the corresponding eigenvector cannot be recovered from the noise. In the asymptotic regime, the value of the threshold can be computed from the noise and the ratio between dimensionality and the number of samples. However, for a finite sample size, the resulting threshold turns out to be too conservative. This will motivate the use of a simple thresholded estimator, where the threshold now is manual parameter which is set during a training stage.

3.1 Empirical Wiener filters for denoising

Often in the denoising literature, there is a single observation available ($n = 1$), which makes the estimation of the *a priori* parameters $\boldsymbol{\mu}$ and $C_{\boldsymbol{x}}$ highly ill-posed. The available observation can be the whole noisy image (as in some Bayesian wavelet shrinkage methods [22]), a noisy patch (as in block DCT methods [47, 58]), or a 3D stack of noisy patches, as in BM3D [15]. These works restrict $\boldsymbol{\mu}$ and $C_{\boldsymbol{x}}$, by assuming that $\boldsymbol{\mu} = 0$ and that the eigenvectors $\boldsymbol{u}_1, \dots, \boldsymbol{u}_d$ of $C_{\boldsymbol{x}}$ are given by a fixed orthonormal basis (such as wavelets, or DCT). In doing so, only the variances λ_i have to be estimated. A common strategy is to estimate them from the unique observation available as $\widehat{\lambda}_i = \max\{\langle y, \boldsymbol{u}_i \rangle^2 - \sigma^2, 0\}$. In addition, some works use a two-step approach. The first step consists in obtaining a rough estimate of $\boldsymbol{x}$ with non-Bayesian methods, such as universal wavelet thresholding. This so-called *pilot* or *basic* estimate is used in the second step to learn the parameters of the *a priori* distribution [15, 22].

More interesting is the case in which several samples are available. Some works fit a Gaussian model (or a mixture of Gaussian models) to a set of n patches [11, 32, 63]. The estimation problem is now "better posed", and estimates of the mean and the covariance matrix (both eigenvectors and eigenvalues) can be provided, at least when n is sufficiently large compared to the dimension d. In our setting, similar to [32], we estimate the mean and covariance from n patches given by a reference patch and its $n - 1$ nearest neighbors.

Since $C_{\boldsymbol{y}} = C_{\boldsymbol{x}} + \sigma^2 I_d$ the authors in [32] proposed to estimate $C_{\boldsymbol{x}}$ as $\widehat{C}_{\boldsymbol{x}} = \widehat{C}_{\boldsymbol{y}} - \sigma^2 I_d$, where $\widehat{C}_{\boldsymbol{y}}$ is the sample covariance matrix of the set of observed noisy patches. While the eigenvalues of $C_{\boldsymbol{y}}$ are larger or equal than σ^2, the estimation errors on $\widehat{C}_{\boldsymbol{y}}$ due to the finite sample size cause small eigenvalues to appear below σ^2.

Denoting by $\widehat{\xi}_i$ the eigenvalues of $\widehat{C}_{\boldsymbol{q}}$, the eigenvalues of $\widehat{C}$ are $\widehat{\lambda}_i = \widehat{\xi}_i - \sigma^2$. The resulting filter coefficient $\widehat{s}_{ii}$ is given by

$$\widehat{s}_{ii} = \frac{\widehat{\lambda}_i}{\widehat{\lambda}_i + \sigma^2} = 1 - \sigma^2 / \widehat{\xi}_i,$$

and can be negative and of arbitrarily large magnitude as $\widehat{\xi}_i$ approaches 0. These large negative weights will amplify the components on the directions of smaller variance, which mostly capture noise, resulting in a poor estimate. To avoid this, it seems reasonable to clip the negative *a priori* variances as $\widehat{\lambda}_i = (\widehat{\xi}_i - \sigma^2)_+$. Such estimate of the covariance matrix has been also used on the context of patch-based denoising in [11].

Under the assumption than the *a priori* distribution is Gaussian a natural estimator is the maximum likelihood (ML) estimator. Assuming that the mean is known, the ML estimator of the inverse covariance matrix $Q_{\boldsymbol{y}} = C_{\boldsymbol{y}}^{-1}$ results from the following convex semidefinite program (for a derivation see [4], §7.1.1):

$$\begin{aligned} &\max \ \log\det Q_{\boldsymbol{y}} - \operatorname{tr}(Q_{\boldsymbol{y}}\widehat{C}_{\boldsymbol{y}}) \\ &\text{subject to} \ \ 0 \prec Q_{\boldsymbol{y}} \preceq \sigma^{-2} I_d. \end{aligned}$$

The inequalities in the constraint force the eigenvalues of $S_{\boldsymbol{y}}$ to be on the interval $(0, \sigma^{-2}]$. The ML estimator for $C_{\boldsymbol{x}}$, results from inverting the solution of the above SDP and subtracting $\sigma^2 I_d$, and is guaranteed to be positive semidefinite. Although there are efficient algorithms for solving such an SDP, they are still prohibitive for the present application.

The covariance matrix estimation has of course many applications other than image restoration, and there is a vast literature in statistics about the topic. Most methods focus on classes of matrices with some sparsity constraint, either on the covariance matrix itself or on its inverse, the precision matrix. See [10] for a recent survey. Among these matrix classes, the one that fits more naturally our local patch models is the spiked covariance, which expresses the covariance matrix as a low-rank signal component plus noise (AGWN). Several authors [18, 28, 45] studied the relation between the sample covariance matrix and the population one, showing that the sample eigenvectors and eigenvalues are inconsistent estimators. Cai et al. [9], impose additionally a group-sparsity constraint on the signal eigenvectors, and proposed estimators with an optimal minimax risk convergence rate. The proposed estimators are mainly of theoretical interest, since require a global search for the support of the eigenvectors.

Here we focus on the simpler problem of estimating the eigenvalues, and propose two estimators derived from the asymptotic results of [18] that leads to improved reconstruction results in comparison to other estimators used previously in the denoising literature, with the same computational cost.

In the remainder of the section we focus our attention on the estimation of the covariance matrix and assume that the mean $\boldsymbol{\mu}$ is zero and it is known or can be well estimated.

3.2 Convergence of the sample covariance matrix

We start by reviewing the results of Paul [18] on the asymptotic behaviour of the spectral decomposition of the sample covariance matrix for *spiked covariance* models. We consider that samples $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$ are Gaussian random variables with $\mathbf{0}$ mean and covariance

$$C_{\boldsymbol{x}} = U \operatorname{Diag}(\lambda_1, \dots, \lambda_m, 0, \dots, 0)\, U^T,$$

with the eigenvalues sorted in non-increasing order. The noisy observations $\boldsymbol{y}_1, \dots, \boldsymbol{y}_n$ are contaminated with white Gaussian noise of standard deviation σ^2. The following result can be obtained from Theorems 1 and 2 in [18] via a change of variables.

Theorem 1 (Almost sure limit of the sample eigenvalues) *Suppose that $d/n \to \gamma \in (0,1)$ as $n \to \infty$. Let $\widehat{\xi}_i$ be the ith largest eigenvector of the sample covariance matrix $\widehat{C}_{\boldsymbol{y}}$, with $1 \leqslant i \leqslant m$. Then $\widehat{\xi}_i$ converges almost surely to*

$$\widehat{\xi}_i \to \begin{cases} \sigma^2(1+\sqrt{\gamma})^2 & \text{if } \lambda_i \leqslant \sqrt{\gamma}\sigma^2, \\ f(\lambda_i) := (\lambda_i + \sigma^2)\left(1 + \dfrac{\gamma\sigma^2}{\lambda_i}\right) & \text{if } \lambda_i > \sqrt{\gamma}\sigma^2. \end{cases}$$

Furthermore, if λ_i has multiplicity one, the scalar product between the corresponding sample eigenvector $\widehat{\boldsymbol{u}}_i$ and the population eigenvector $\boldsymbol{u}_i$ converges almost surely to

$$\langle \widehat{\boldsymbol{u}}_i, \boldsymbol{u}_i \rangle^2 \to \begin{cases} 0 & \text{if } \lambda_i \leqslant \sqrt{\gamma}\sigma^2, \\ \dfrac{1-\gamma\sigma^4/\lambda_i^2}{1+\gamma\sigma^2/\lambda_i} & \text{if } \lambda_i > \sqrt{\gamma}\sigma^2. \end{cases}$$

The interesting fact here is that there is a threshold $\sqrt{\gamma}\sigma^2$ below which the eigenvalue-eigenvector pair cannot be recovered from the noise. The squared root of the ratio γ between the dimensionality and the number of samples acts as a noise amplifier in the threshold. As intuition suggests, the lower the number of samples in relation to the dimensionality, the higher the threshold for recoverability.

Additionally, it is shown in [18] that the recoverable sample eigenvalues and eigenvectors are asymptotically normal.

3.3 Estimators for the eigenvalues of $C_{\boldsymbol{x}}$

We can derive an estimator for λ_i by assuming that the asymptotic regime holds. The population eigenvalue λ_i and the corresponding eigenvector are recoverable if and only if $\xi > \sigma^2(1+\sqrt{\gamma})^2$. This gives us a recoverability threshold on the sample eigenvalue. If a sample eigenvalue is below that threshold, it is reasonable to set the population eigenvalue to zero, since the corresponding eigenvector is essentially noise.

The function f is invertible when restricted to $[\sqrt{\gamma}\sigma^2, \infty)$. Therefore, if $\widehat{\xi}_i$ is above the recoverability threshold, we can estimate it as $\widehat{\lambda}_i = f^{-1}(\widehat{\xi}_i)$. We thus propose the following estimator $\widehat{\lambda}^{\mathrm{S}}$:

$$\widehat{\lambda}^{\mathrm{S}}(\widehat{\xi}) = \begin{cases} 0 & \text{if } \widehat{\xi} \leqslant \sigma^2(1+\sqrt{\gamma})^2, \\ f^{-1}(\widehat{\xi}_i) & \text{if } \widehat{\xi} > \sigma^2(1+\sqrt{\gamma})^2. \end{cases}$$

where

$$f^{-1}(\widehat{\xi}_i) = \frac{1}{2}\left(1 + \sqrt{1 - \frac{4\gamma\sigma^4}{(\widehat{\xi} - \sigma^2(1+\gamma))^2}}\right).$$

Figure 3 shows a plot of $\widehat{\lambda}^{\mathrm{S}}(\xi)$ for $\sigma = 5$ and $\gamma = 1$. When $\widehat{\xi} \gg \sigma^2(1+\sqrt{\gamma})^2$, the estimator approaches $\widehat{\xi} - \sigma^2(1+\gamma)$. As the estimator approximates a linear function of $\widehat{\xi}$, the bias reduces, since

$$\mathbb{E}\{\widehat{\lambda}^{\mathrm{S}}(\widehat{\xi})\} \approx \mathbb{E}\{f^{-1}(\widehat{\xi}_i)\} \approx f^{-1}(\mathbb{E}\{\widehat{\xi}_i\}) = f^{-1}(f(\lambda_i)) = \lambda_i.$$

The first approximation is because when λ_i is large enough, the probability that $\widehat{\xi}$ falls below the recoverability threshold is negligible. And the second approximation is due to the approximate linearity of the estimator.

The proposed estimator is valid if the asymptotic regime holds, but we have no optimality guarantees for a finite size sample. Nadler [45] provides finite sample size bounds but only for the case $m = 1$. In practice for our video denoising application, we use patches with dimensionality between $d = 100$ and $d = 200$ (for example for a patch of $5 \times 5 \times 4$ or $7 \times 7 \times 4$). Typical sample sizes are of the same order.

In Figure 4 we show results of experiments with simulated data. We consider $m = 20$ eigenvalues $\lambda_1 = 20, \lambda_2 = 19, \ldots, \lambda_m = 1$, and noise with $\sigma = 2$. We show the estimated eigenvalues (averaged over 400 simulations) for different values of the dimensionality d and the ratio γ. As expected, both estimators perform better for smaller γ, since the number of samples is higher. The

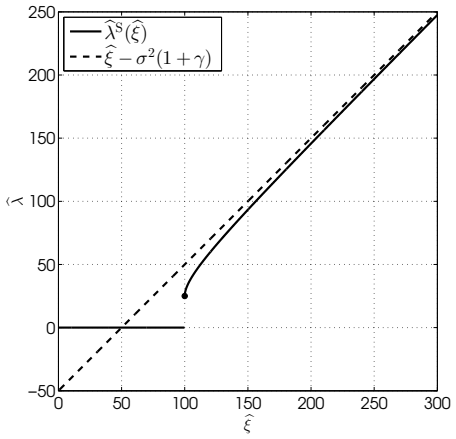

Fig. 3: Proposed variance estimator $\widehat{\lambda}^{\mathrm{S}}$ as a function of the sample eigenvalue $\widehat{\xi}$, for $\gamma = 1$ and $\sigma = 5$. If $\xi \gg \sigma^2(1+\sqrt{\gamma})^2$, then $\widehat{\lambda}^{\mathrm{S}}$ approximates $\widehat{\xi} - \sigma^2(1+\gamma)$.

estimator $(\hat{\xi} - \sigma^2)_+$ has a positive bias. When γ increases the bias increases for the first eigenvalues. Although it drops faster to zero, it still largely overestimates many zero eigenvalues, resulting in a filter which allows many noise components to pass. When d increases, the bias on the m first eigenvectors reduces, but on the other hand, a larger number of zero eigenvalues are estimated as positive. The estimator $\widehat{\lambda}^{\mathrm{S}}$ is much more accurate. The results in the first row confirm that when the asymptotic regime applies, the estimator behaves well. For larger values of γ and lower dimensionality, the estimation accuracy degrades, and many positive eigenvalues are estimated as close to zero, even if they are above the recoverability threshold.

In the light of this, we propose an additional estimator by hard thresholding the difference $\hat{\xi} - \sigma^2$. The value of the threshold is given by a parameter τ:

$$\hat{\lambda}_i^{\mathrm{H}} = \mathrm{H}_\tau(\hat{\xi}_i - \sigma^2) = \begin{cases} \hat{\xi} - \sigma^2 & \text{if } \hat{\xi} \geqslant \tau\sigma^2 \\ 0 & \text{if } \hat{\xi} < \tau\sigma^2. \end{cases}$$

For $\tau = 0$ we recover $(\hat{\xi} - \sigma^2)_+$. In this way, varying τ we can balance between overestimating zero eigenvalues and underestimating signal components. Determining the optimal value for τ for a finite sample size is a difficult task. Instead, we shall include it as an additional parameter of the denoising algorithm and tune it over a training dataset.

Our final objective is not estimating the eigenvalues of the *a priori* covariance matrix, but the clean data vectors from the noisy ones. Therefore, we evaluate the performance of each estimator by measuring the denoising MSE obtained by the resulting empirical Wiener filter. Figure 5 shows the denoising MSE obtained by different estimators of the spectrum of $C_{\boldsymbol{x}}$: $(\hat{\xi} - \sigma^2)_+$, $\widehat{\lambda}^{\mathrm{S}}$ and $\hat{\lambda}_\tau^{\mathrm{H}}$, varying the value of $\tau > 0$. The MSE is estimated as an average of the denoising errors obtained for 400 realizations of the noisy dataset. The obtained MSE is normalized by the minimum MSE and is plotted as a function of τ. Each plot corresponds to a different value of the dimensionality d and the ratio γ as in Figure 4. Additionally, we also show the MSE of an oracular empirical Wiener filter built from the true population variances λ but using the eigenvectors $\widehat{\boldsymbol{u}}_i$ of the sample covariance matrix.

The best MSE attained by the empirical Wiener filters depends mainly on the ratio γ. As expected, when n decreases in relation with the dimensionality, the denoising MSE increases. In most cases, the eigenvalues $\widehat{\lambda}^{\mathrm{S}}$ yield the best results, close to the ones obtained with the oracular

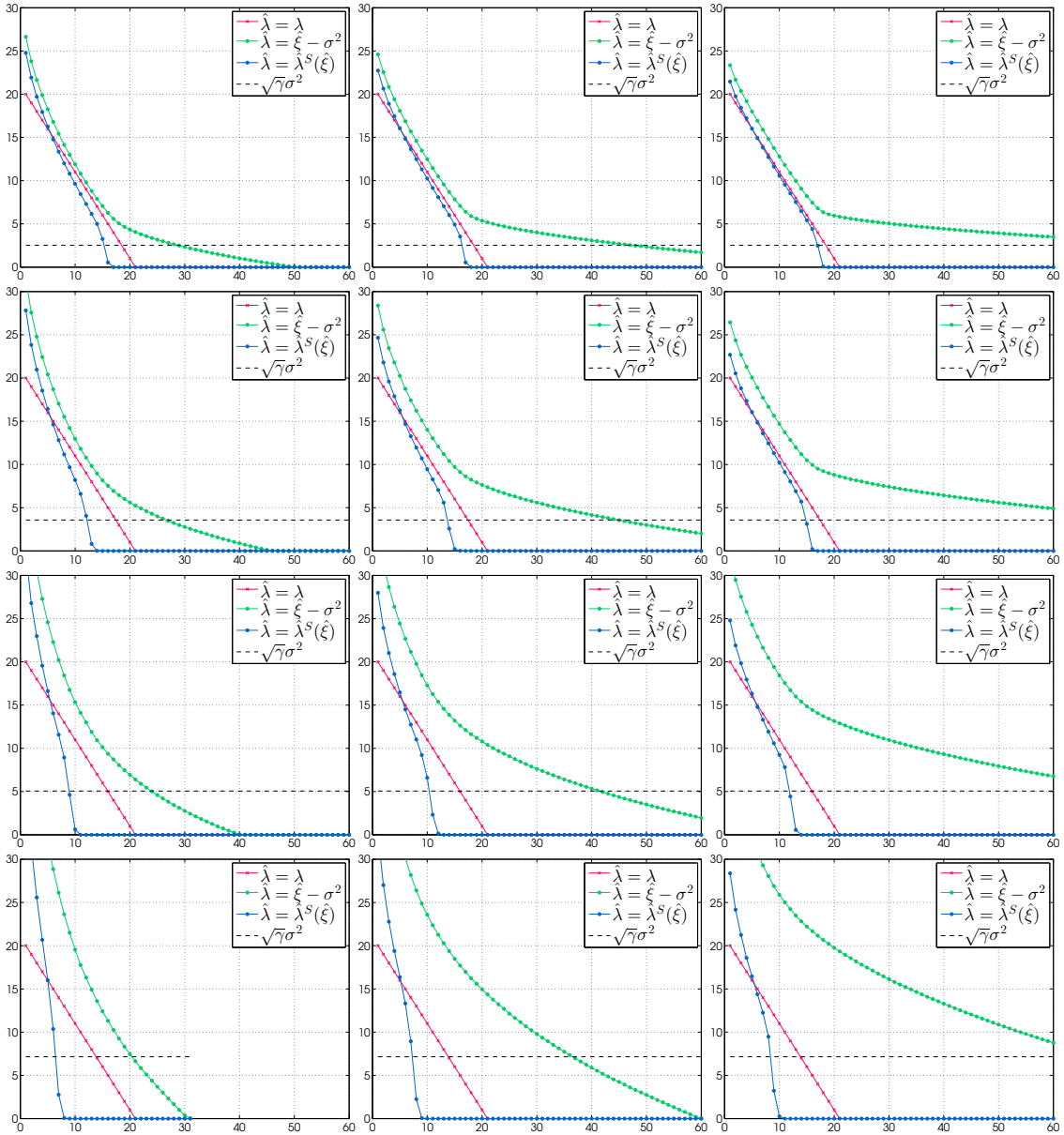

Fig. 4: Estimated variances when varying the dimensionality d and the ratio γ. Each plot shows the true variances, and the ones estimated from the sample covariance matrix of the noisy data using two estimation methods. The horizontal axis corresponds to the eigenvalue index $i = 1, \ldots, 60$, and the vertical axis to the value of the estimated eigenvalue using the different estimators. From left to right, results with $d = 100, 200$ and 400. From top to bottom, $\gamma = 0.4, 0.8, 1.6$ and 3.2. Note that diagonals have the same number of samples $n = d/\gamma$.

λ, and even slightly better for $d = 400$ and $\gamma = 0.2$.[4] As expected its denoising performance drops when d is small and γ large. In these situations, better results are obtained with $(\hat{\xi} - \sigma^2)_+$. The results obtain with the estimator $\widehat{\lambda}^{\mathrm{H}}$ depend on τ, but the optimal value is always close to the result attained by the best among the other filters.

4 Parameter selection

In view of the results of the previous section, we shall consider two variants of the video NL-Bayes algorithm described in Algorithms 1 and 2 which differ on the estimator used for the eigenvalues of the *a priori* covariance matrix. We denote by VNLB-S the version of the algorithm that uses the aymptotic estimator $\widehat{\lambda}^{\mathrm{S}}$, and by VNLB-H the one corresponding to the thresholded estimator $\widehat{\lambda}^{\mathrm{H}}$.

For each stage of the algorithm we need to specify the following parameters: the spatial and temporal components of the patch size s_x and s_t, the spatial and temporal components of the search region w_x and w_t, and the number of similar patches n. In addition, for VNLB-H we also need to specify the value of τ. When needed, we will add a subscript indicating the stage, *e.g.* n_1 and n_2. We consider different parameter settings for color and grayscale videos. [5]

The parameters of the search region are chosen manually. The rest of the parameters are determined by sampling the parameter space and picking the ones which maximize the average PSNR over a training set of sequences. We consider a grayscale training set containing six sequences of twenty frames, and a color training set of four sequences of ten frames. We selected training sequences having different characteristics on their dominant motion (static, translational, non-rigid) and on their image content (random textures or geometric structures).

4.1 Search region

Due to the temporal consistency of natural image sequences, the patches similar to the reference patch are expected to be located around its motion trajectory in neighboring frames. Thus we define the spatio-temporal search region as a sequence of w_t square windows of size $w_x \times w_x$, whose centers follow the motion trajectory of the reference patch. To estimate the trajectory we compute the forward and backward optical flows of the sequence. The forward half of the trajectory $\varphi_{x,t}$ passing throught (x, t) is computed by integrating the forward optical flow v^f as follows:

$$\varphi_{x,t}(h) = v^f([\varphi_{x,t}(h-1)], h-1) + \varphi_{x,t}(h-1), \quad h = t+1, \ldots, t + \lfloor w_t/2 \rfloor,$$

where $[\,\cdot\,]$ and $\lfloor\cdot\rfloor$ denote the round and floor operators. The backward half of the trajectory is defined analogously using the backward optical flow v^b.

In all our experiments we set $w_x = 27$, and $w_t = 13$, yielding a search volume of 9477 patches. Due to the size of the spatial search window, the optical flow does not need to be accurate. We have tested with two optical flow algorithms: the TV-L1 method [60] (we used the implementation of [53]), and its robust variant described [43], and have found virtually no difference in the denoising result. There is also no significant difference between computing the optical flow on the noisy data, or using an oracular optical flow computed on the ground truth. For our experiments, we use the TV-L1 optical flow since it is fast to compute.

[4] It might seem counterintuitive that the filter computed from the estimated eigenvalues $\widehat{\lambda}^{\mathrm{S}}$, outperforms the oracular one. It is a consequence of the error in the eigenvectors. In particular, eigenvectors corresponding to eigenvalues below the recoverability threshold are better discarded.

[5] In order to reduce the parameter space, we did not optimize the choice of the distance threshold ρ.

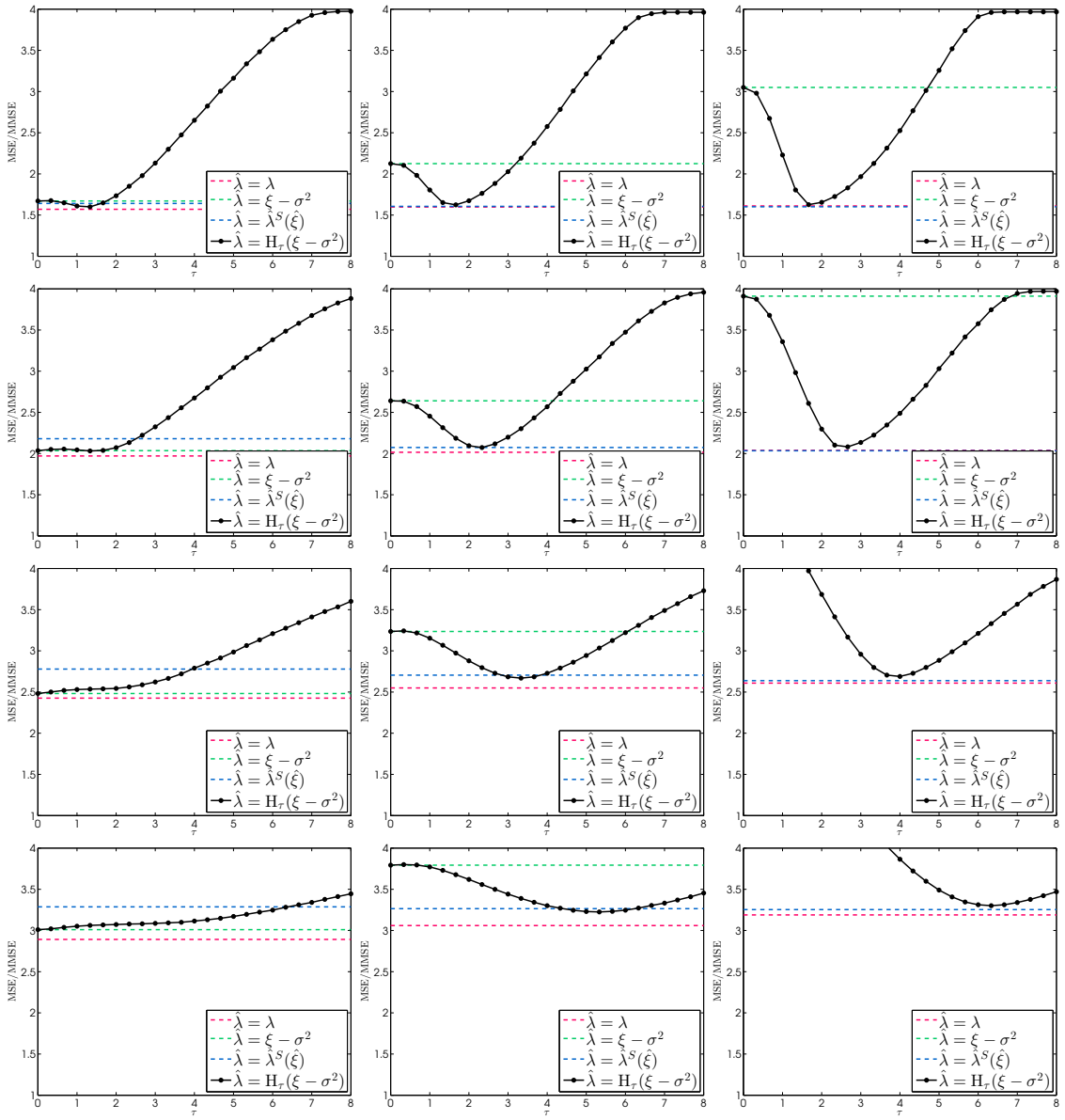

Fig. 5: MSE obtained by estimating the n data samples from their noisy observations using empirical Wiener filters with different estimators for the *a priori* variances λ. The plots correspond to ones in Figure 4. The MSE is normalized by the minimum MSE (attained by the Wiener filter) and it is shown as a function of the parameter τ.

4.2 Parameters of Bayesian estimation

The empirical Bayesian estimation is controlled by the number of patches n and the parameters of the variance estimator. We determine these parameters as functions of the noise level σ for each of the patch sizes considered.

Based on the results of §3 on simulated data, using a higher number of samples to estimate the *a priori* covariance matrix yields a lower MSE. In the context of our denoising algorithm, as we increase the number n of similar patches, we include patches that are not as similar to the reference patch and the assumption that the selected patches are IID samples becomes less valid. Therefore there is trade-off between the error in the sample covariance matrix and the accuracy of the model.

The thresholded eigenvalue estimator used in the VNLB-H variant requires the specification of the additional parameter τ. For a fixed patch size, there is an interplay between n and τ. This can be intuitively understood in the light of the asymptotic theory, where the recoverability threshold depends on the ratio γ between the number of samples and the dimensionality. There is also another interplay between the parameters of both stages n_1, τ_1 and n_2, τ_2. Due to this coupling we sample this four dimensional parameter space and search exhaustively for the parameters that optimize the mean PSNR on the training set, for different values of σ. To simplify (and avoid over-fitting) we restrict the relationship between the parameters and σ to be linear.

The asymptotic eigenvalue estimator $\widehat{\lambda}^{\mathrm{S}}$ used in the VNLB-S variant does not require the additional parameter τ, but depends on the noise level σ. This estimator is based on Paul's asymptotic theory [18] which applies only for low-rank covariance matrices when the observations are contaminated with AGWN. This suits well to the situation in the first step of the algorithm, where the covariance is estimated using the noisy patches. This is not the case in the second step of the algorithm, where patches of the basic estimate are used. The application of Paul's theory in this case is rather forced, since the residue in the basic estimate is not AGWN. In spite of this, a soft-thresholding of the eigenvalues in our estimator seems reasonable under the assumption that the similar patches should lie close to a lower dimensional space. In this case we can think of $\widehat{\lambda}^{\mathrm{S}}$ as a parametric estimator, such as a $\widehat{\lambda}^{\mathrm{H}}_{\tau}$. In the first step, since the noise AGWN, the theory in [18] gives the appropriate value of the parameter as the noise level σ. For the second step, we found *empirically* that an effective way to set the noise parameter is to assign an artificial noise level to the basic estimate, denoted as $\sigma^{(1)}$ in Algorithms 1 and 2. We estimate this artificial noise level with the noise estimation algorithm [12].[6] The remaining parameters n_1 and n_2, are determined by a 2D parameter search on the training set. As before, we allow for a linear dependency with the noise level σ.

4.3 Patch size

The patch size is a key parameter of patch-based denoising algorithms [7,30]. Increasing the patch size reduces the variance on the patch distance and therefore on the set of similar patches. However, larger patches are less sensitive to finer scale details and less similar among them. We solved this trade-off empirically, by testing different patch sizes on a set of training sequences.

We consider patches sizes with temporal extent ranging from 1 to 4 frames, and dimensionality $d \approx 100$ and $d \approx 200$:

$$\begin{array}{lllll} d \approx 100: & 10\times 10\times 1 & 7\times 7\times 2 & 6\times 6\times 3 & 5\times 5\times 4, \\ d \approx 200: & 14\times 14\times 1 & 10\times 10\times 2 & 8\times 8\times 3 & 7\times 7\times 4. \end{array}$$

[6] We use the default parameters in [12], with the exception of the number of bins, which we set to one since we are not interested in a noise curve, but in just an average noise level.

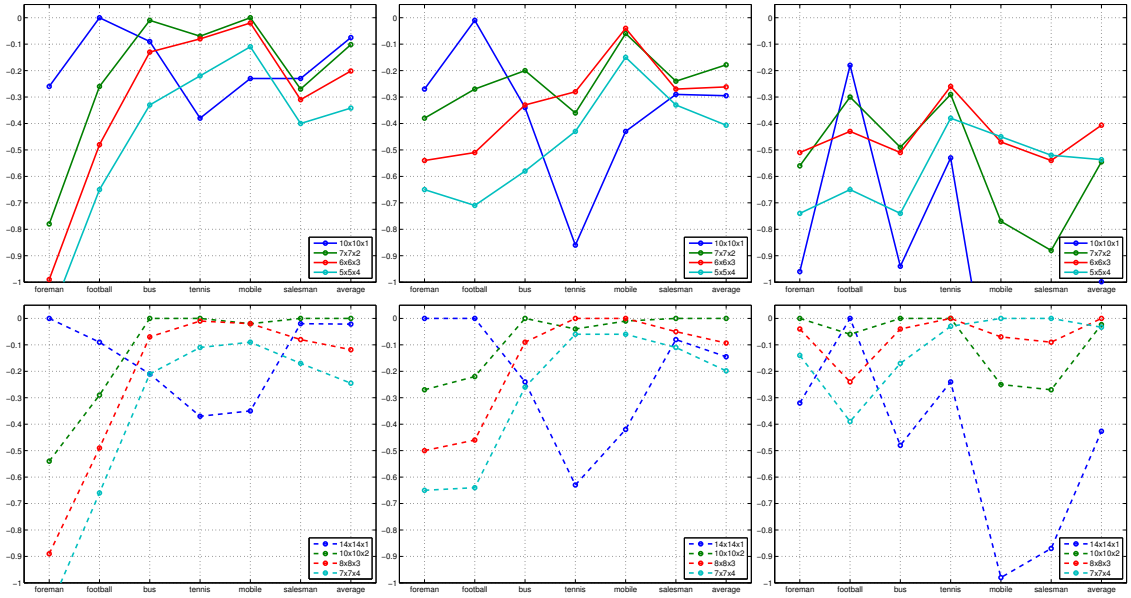

Fig. 6: Highest attainable PSNR for each patch size over the grayscale training set. For each patch size and sequence, we plot on the vertical axis the highest PSNR obtained with that patch size for that sequence minus the highest PSNR value attained for the sequence among all considered patch sizes. The rightmost points on each plot correspond to the average PSNR over the training set. The columns correspond to the noise levels $\sigma = 6, 12, 48$. The top row shows the results for patches sizes with $d \approx 100$ and those with $d \approx 200$ are shown in the bottom row.

	Method	w_x	w_t	n_1	n_2	τ_1	τ_2
Gray	VNLB-H $7^2 \times 2$	27	6	150	$[42.9 - .48\sigma]$	2.1	$\max\{0.5, 2.53 - .056\sigma\}$
	VNLB-H $10^2 \times 2$	27	6	150	60	3.7	$\max\{0.0, 1.87 - .028\sigma\}$
	VNLB-S $7^2 \times 2$	27	6	60	60	n/a	n/a
	VNLB-S $10^2 \times 2$	27	6	100	$[86.7 - .56\sigma]$	n/a	n/a
RGB	VNLB-S $7^2 \times 2$	27	6	100	$\max\{30, 80 - 2\sigma\}$	n/a	n/a
	VNLB-S $10^2 \times 2$	27	6	100	$100 - \sigma$	n/a	n/a

Table 1: Parameters used for all results shown in §5, for the two variants of the method and two patch sizes. The parameters w_x and w_t are the same for both steps. For RGB sequences we only consider the VNLB-S method, since its parameter space is smaller.

In Figure 6 we plot for each patch size the best attainable PSNR on each sequence of the grayscale training set. The larger patches with $d \approx 200$ show better performance, specially as the noise level increases. We can also see that for sequences with stronger motion (such as *football* and *foreman*) it is better to use patches with a shorter time span, of two and even one frame. On the other hand on more static sequences, *e.g. mobile* and *salesman*, it is better to use a larger time span. For both dimensionalities, patches with two and three frames perform well for the tested noise levels. For low noise levels 2D patches perform well, but their performance decays drastically when σ increases.

Based on these results, in the following we will consider two patch sizes: $7 \times 7 \times 2$ and $10 \times 10 \times 2$. The corresponding optimal parameters are shown in Table 1.

5 Results

In this section we present results obtained with the proposed methods on some classical test sequences with white Gaussian noise added. We consider separately the cases of grayscale and color videos. For grayscale sequences we show results obtained with both variants of our method, VNLB-H and VNLB-S, with patch sizes $7 \times 7 \times 2$ and $10 \times 10 \times 2$. These attained the best average performance on the training sequences among the other patch sizes of the same dimensionality. We used the parameters of Table 1. For color sequence we only show the results of VLNB-S. Some sample sequences can be downloaded from the project website[7] in uncompressed format.

σ	Method	Tennis	Sales.	Garden	Mobile	Bicycle	Stefan	Average	
10	V-BM4D-tip	35.22	37.30	32.81		37.66			
	V-BM4D-mp	34.95	37.48	32.01	34.11	37.85	33.68	*34.72*	35.01
	VNLB-S $7^2 \times 2$	35.65	38.02	34.05	36.21	39.39	35.42	*36.15*	36.46
	VNLB-S $10^2 \times 2$	35.89	38.36	34.42	36.37	39.37	35.83	*36.39*	36.71
	VNLB-H $7^2 \times 2$	35.73	38.23	34.47	**36.63**	**39.62**	35.78	*35.10*	36.74
	VNLB-H $10^2 \times 2$	**36.00**	**38.62**	**34.61**	**36.67**	39.53	**36.02**	*35.94*	**36.91**
20	V-BM4D-tip	31.59	33.79	28.63		34.10			
	V-BM4D-mp	31.08	33.46	28.32	30.49	34.54	29.69	*30.53*	31.26
	VNLB-S $7^2 \times 2$	32.01	34.39	30.00	32.26	36.20	31.39	*31.93*	32.71
	VNLB-S $10^2 \times 2$	32.14	34.73	30.24	32.42	36.22	31.70	*32.39*	32.91
	VNLB-H $7^2 \times 2$	32.00	34.55	30.26	32.56	**36.42**	31.60	*30.67*	32.90
	VNLB-H $10^2 \times 2$	**32.20**	**34.99**	**30.46**	**32.82**	**36.45**	**31.94**	*31.77*	**33.14**
30	V-BM4D-tip	29.72	31.75	26.29		31.83			
	V-BM4D-mp	29.37	31.02	26.20	27.99	32.30	27.34	*28.03*	29.04
	VNLB-S $7^2 \times 2$	30.19	32.10	27.61	29.73	33.94	29.00	*29.26*	30.43
	VNLB-S $10^2 \times 2$	30.29	32.54	27.85	30.03	34.12	29.36	*29.90*	30.70
	VNLB-H $7^2 \times 2$	30.26	32.27	27.66	29.94	34.15	29.05	*28.07*	30.56
	VNLB-H $10^2 \times 2$	**30.37**	**32.77**	**28.04**	**30.41**	**34.37**	**29.56**	*29.33*	**30.92**
40	V-BM4D-tip	28.49	30.35	24.60		30.10			
	V-BM4D-mp	28.38	29.37	24.59	26.02	30.58	25.64	*26.22*	27.43
	VNLB-S $7^2 \times 2$	29.02	30.32	25.86	27.79	32.16	27.20	*27.28*	28.73
	VNLB-S $10^2 \times 2$	29.14	30.89	26.18	28.24	32.44	27.71	*28.03*	29.10
	VNLB-H $7^2 \times 2$	29.02	30.49	25.69	27.92	32.31	27.19	*26.19*	28.77
	VNLB-H $10^2 \times 2$	**29.24**	**31.09**	**26.29**	**28.55**	**32.70**	**27.84**	*27.57*	**29.29**

Table 2: Quantitative denoising results for some classic grayscale test sequences. The values in *italics* correspond to the average of the PSNR of the basic estimates. See text for details.

5.1 Results on grayscale videos

We present denoising results on grayscale test sequences used commonly in the literature. The sequences have CIF or SIF resolution and between 150 and 300 frames.

[7] `http://dev.ipol.im/~pariasm/video_nlbayes/`

		Grayscale sequences					Color sequences				
σ	Method	Bus	Tennis	Sales.	Bike	Ave.	Army	Cooper	Dog	Truck	Ave.
10	SPTWO	36.07	34.69	36.38	36.74	35.97	39.44	36.09	35.90	37.10	37.13
	VNLB-S $7^2 \times 2$	36.32	34.67	38.13	39.25	37.09	39.30	**37.31**	36.66	**37.29**	37.64
	VNLB-S $10^2 \times 2$	36.68	34.86	38.51	39.22	37.32	**39.62**	**37.31**	**36.83**	**37.31**	**37.77**
	VNLB-H $7^2 \times 2$	36.68	35.04	38.31	**39.45**	37.37					
	VNLB-H $10^2 \times 2$	**36.98**	**35.18**	**38.77**	39.38	**37.58**					
20	SPTWO	32.24	30.59	32.95	33.01	32.20	**36.47**	32.34	32.94	33.65	33.85
	VNLB-S $7^2 \times 2$	32.31	30.35	34.44	36.09	33.30	36.06	33.44	33.80	**33.75**	34.26
	VNLB-S $10^2 \times 2$	32.77	30.70	34.78	36.11	33.59	36.28	**33.49**	**33.92**	**33.76**	**34.36**
	VNLB-H $7^2 \times 2$	32.52	30.29	34.61	36.28	33.42					
	VNLB-H $10^2 \times 2$	**33.08**	**30.91**	**35.06**	**36.32**	**33.84**					
30	SPTWO	30.05	27.48	30.95	31.62	30.03	**34.67**	30.24	**31.29**	31.61	31.95
	VNLB-S $7^2 \times 2$	29.98	27.69	32.15	33.85	30.92	34.26	31.28	32.13	**31.74**	32.35
	VNLB-S $10^2 \times 2$	30.50	27.91	32.57	34.04	31.25	34.52	**31.38**	**32.32**	**31.79**	**32.50**
	VNLB-H $7^2 \times 2$	30.11	27.64	32.33	34.05	31.03					
	VNLB-H $10^2 \times 2$	**30.76**	**27.84**	**32.83**	**34.30**	**31.43**					
40	SPTWO	28.51	25.53	29.60	29.77	28.35	**33.24**	28.85	30.15	30.19	30.61
	VNLB-S $7^2 \times 2$	28.21	26.10	30.40	32.12	29.21	32.90	29.78	30.88	30.35	30.98
	VNLB-S $10^2 \times 2$	28.76	26.32	30.98	32.42	29.62	**33.29**	**29.94**	**31.19**	**30.44**	**31.22**
	VNLB-H $7^2 \times 2$	28.28	26.05	30.57	32.30	29.30					
	VNLB-H $10^2 \times 2$	**28.97**	**26.22**	**31.19**	**32.71**	**29.77**					
50	SPTWO	27.19	24.96	28.34	28.66	27.29	31.99	27.78	29.09	29.13	29.50
	VNLB-S $7^2 \times 2$	26.95	25.07	29.03	30.59	27.91	31.82	28.70	29.91	29.27	29.92
	VNLB-S $10^2 \times 2$	27.56	25.20	29.67	31.00	28.36	**32.32**	**28.93**	**30.30**	**29.43**	**30.24**
	VNLB-H $7^2 \times 2$	27.05	25.00	29.20	30.82	28.02					
	VNLB-H $10^2 \times 2$	**27.71**	**25.13**	**29.86**	**31.25**	**28.49**					

Table 3: Comparison with the method SPTWO. Shown PSNRs correspond to the central frame for each test sequences. See text for details.

σ	Method	Tennis	Sales.	Garden	Mobile	Bicycle	Stefan	Average
10	VNLB-H $7^2 \times 2$	35.73	38.23	34.47	36.63	39.62	35.78	36.74
	VNLB-H $7^2 \times 2$ (no m.c.)	35.71	38.23	34.29	36.60	39.51	35.06	36.59
20	VNLB-H $7^2 \times 2$	32.00	34.55	30.26	32.56	36.42	31.60	32.90
	VNLB-H $7^2 \times 2$ (no m.c.)	31.98	34.55	30.01	32.53	36.27	30.74	32.71
30	VNLB-H $7^2 \times 2$	30.26	32.27	27.66	29.94	34.15	29.05	30.56
	VNLB-H $7^2 \times 2$ (no m.c.)	30.23	32.26	27.40	29.91	33.96	28.23	30.36
40	VNLB-H $7^2 \times 2$	29.02	30.49	25.69	27.92	32.31	27.19	28.77
	VNLB-H $7^2 \times 2$ (no m.c.)	28.99	30.50	25.44	27.89	32.10	26.46	28.59

Table 4: Results without a motion compensated search region (no m.c.). See text for details.

σ	Method	Tennis	Coast.	Fore.	Bus	Foot.	Average	
5	V-BM3D	39.45	40.18	40.16	39.07			39.71
	V-BM4D-tip	39.98	41.13	41.38	40.21			40.68
	V-BM4D-mp	39.60	40.28	40.78	39.56	40.00	*39.98*	40.06
	VNLB-S $7^2 \times 2$	**40.29**	**41.43**	41.89	**41.27**	**41.23**	*41.13*	**41.22**
	VNLB-S $10^2 \times 2$	39.92	41.26	**42.13**	41.05	40.89	*41.14*	41.09
10	V-BM3D	36.04	36.82	37.52	34.96			36.34
	V-BM4D-tip	36.42	37.27	37.92	36.23			36.96
	V-BM4D-mp	35.90	36.30	37.21	35.38	36.08	*35.91*	36.20
	VNLB-S $7^2 \times 2$	36.70	37.78	38.64	37.65	37.50	*37.46*	37.69
	VNLB-S $10^2 \times 2$	**36.88**	**38.01**	**39.05**	**37.94**	**37.62**	*37.71*	**37.97**
15	CIFIC	32.76			33.47	31.59		
	V-BM4D-mp	33.67	34.05	35.17	33.02	33.82	*33.49*	33.98
	VNLB-S $7^2 \times 2$	34.62	35.69	36.76	35.54	35.34	*35.25*	35.65
	VNLB-S $10^2 \times 2$	**34.85**	**35.95**	**37.13**	**35.91**	**35.51**	*35.59*	**35.96**
20	V-BM3D	32.54	33.39	34.49	31.03			32.86
	V-BM4D-tip	32.88	33.61	34.62	32.27			33.35
	V-BM4D-mp	31.98	32.44	33.70	31.34	32.22	*31.76*	32.37
	VNLB-S $7^2 \times 2$	33.17	34.26	35.43	34.03	33.82	*33.66*	34.22
	VNLB-S $10^2 \times 2$	**33.35**	**34.50**	**35.78**	**34.42**	**33.99**	*34.06*	**34.51**
25	CIFIC	29.74			30.48	28.82		
	V-BM4D-mp	30.73	31.24	32.59	30.07	30.96	*30.44*	31.16
	VNLB-S $7^2 \times 2$	32.07	33.15	34.35	32.81	32.62	*32.42*	33.10
	VNLB-S $10^2 \times 2$	**32.18**	**33.40**	**34.74**	**33.24**	**32.80**	*32.86*	**33.39**
40	V-BM3D	29.20	29.99	31.17	27.34			29.43
	V-BM4D-tip	29.52	30.00	31.30	28.32			29.78
	V-BM4D-mp	28.14	28.73	30.09	27.44	28.35	*27.66*	28.60
	VNLB-S $7^2 \times 2$	29.78	30.89	32.07	30.28	30.15	*29.87*	30.76
	VNLB-S $10^2 \times 2$	**29.92**	**31.19**	**32.60**	**30.69**	**30.36**	*30.34*	**31.10**

Table 5: PSNRs obtained for the classic color test sequences. The averages corresponds to the first four sequences. The values in *italics* correspond to the average of the PSNR of the basic estimates. See text for details.

In Table 2 we compare our results against the state-of-the-art method V-BM4D [39] with two different choices of parameters. The method labeled V-BM4D-tip corresponds to the results reported in [39]. The results labeled V-BM4D-mp were obtained using the implementation available online [40]. This implementation provides three parameter profiles of increasing quality and complexity: "low complexity profile", "normal profile" and "modified profile". We show the results obtained with the modified profile.

Each sequence was contaminated with AGWN of $\sigma = 10, 20, 30, 40$. Table 2 shows the PSNR values obtained after denoising. We can observe that using a larger patch size gives better results, specially for higher noise levels, but at a higher computational cost. For our methods and V-BM4D-mp we show the average PSNR of the basic estimate as well. This allows to see the improvement achieved by the second iteration.

In Table 3 we also include a comparison with SPTWO [8].To denoise each frame, SPTWO uses the optical flow to register the neighboring frames (6 past and future frames) to the target frame.

For each 5×5 reference patch in the target frame, $13k$ similar patches are selected by searching for the k most similar $5 \times 5 \times 13$ patch trajectories in the volume defined by the warped frames. Because patches are warped according to the motion, SPTWO is in principle able to better handle zooms, rotations and non-rigid transformations. This comes at the price of a higher dependency on the optical flow. To compare our method with SPTWO, we computed the result of our algorithm on some of the sequences used in [8]. The grayscale scale sequences have only 30 frames, and the color sequences have 8. [8]

Both variants of our algorithm achieve comparable results. The performance of VNLB-H is slighly better (at most 0.3dB on the average). This is not surprinsing given that VNLB-H uses a hand tuned estimator (on both steps of the algorithm) optimized over the test sequences. With two parameters less, the results of VNLB-S are only slightly inferior. This supports the choice of the basic noise level $\sigma^{(1)}$ to set the noise parameter of the soft-threshold estimator $\widehat{\lambda}^{\mathrm{S}}$ in the second step of the algorithm.

Our results outperform the ones of V-BM4D for all sequences and noise levels, and with the two variants VNLB-S and VNLB-H and patch sizes. On average, the difference with V-BM4D-mp is 1.8dB. The proposed methods also outperform SPTWO on the average (with a margin varying between 1.2dB and 1.6dB depending on the noise level), but the difference is much smaller for *bus* and *tennis*. In these sequences the result of SPTWO is slighly better that the ones we obtain with the patch size of $7 \times 7 \times 2$, except for the highest noise levels.

For a qualitative comparison, we show in Figures 7, and 8, details of the results obtained with the proposed VNLB-H method, V-BM4D-mp and SPTWO,[9] with noise levels of $\sigma = 10, 40$. For VNLB-H we show results obtained with a patch size of $10 \times 10 \times 2$. We omit the results obtained with the VNLB-S since they are visually hard to distinguish.

The differences between the methods become more apparent for higher noise levels. The VNLB-H preserves more details than V-BM4D. This can be noticed for example for *mobile* and *bicycle*. SPTWO achieves very good visual results in moving objects for which the motion can be well approximated. This can be seen in the face of *salesman*. Its results are temporally consistent with the estimated motion. This creates artifacts when the motion is not well estimated. As an example, the wallpaper texture in *tennis* adopts the motion of the arm. This dragging effect is common close to the boundaries of moving objects, particularly if the background texture is weaker than the noise level. In the *bicycle* sequence, for noise $\sigma = 40$ the rays of the wheel are not well reconstructed and the sticker depicting a train shows deformations. This is probably also due to problems with the motion estimation. Notice also that the result of the proposed method is smoother for constant regions, such as the dark background in *bicycle*.

The results shown in Figures 7 and 8 show the spatial characteristics of the proposed method. As a way of assessing the temporal consistency of the result we show a slice of the video volume at a fixed row (or column). Figure 10 shows a horizontal slice of the *mobile* sequence. Each row in the images shown corresponds to a different frame. The gray tube corresponds to a ball which is hit by a black train entering from the left. The result of V-BM4D shows a high frequency variation in the temporal (vertical) dimension, a consequence of the flickering on the video sequence. On the other hand, the result of the video non-local Bayes method, shows a much slower variation in time.

To the best of our knowledge, the best denoising results in grayscale sequences have been reported in [36]. Some of these results are surprising compared to the rest of the state of the

[8] In [8] the error is measured as the root mean square error (RMSE) of the central frame. For color sequences the average channel RMSE is used. To allow a direct comparison, we adopt in Table 3 these error measurements, expressing them as a PSNR in decibels. For color sequences, this amounts to $\mathrm{PSNR} = 20 \log_{10} \left(255/\frac{1}{3}(\mathrm{RMSE}_R + \mathrm{RMSE}_G + \mathrm{RMSE}_B)\right)$.

[9] We are thankful to the authors of [8] for kindly sharing with us their results.

art. For example, for *salesman* with $\sigma = 40$ the result of [36] is 4dB higher than the best result among all public state-of-the-art methods, including ours. We have not included these results in the comparison table since the code is not available and we have not been able to reproduce the results. This method builds upon BM4D, which in [41] is applied to volumetric imagery and video, showing an inferior performance than V-BM4D. The authors in [36] consider shape adaptive patches as in [21] (the patches are quite large $8 \times 8 \times 8$), and perform several iterations. In each iteration, the previous iterate is used as an oracle to compute patch distances and the adaptive shape of the patches. Our method can be iterated as in [36], and some improvement can be obtained in a third iteration, but not enough to compensate for the increase computational cost.

Finally in Table 4 we compare the PSNR obtained on the grayscale sequences of Table 2 without using a motion compensated search region (or equivalently, assuming zero motion). The resulting search region is a $27 \times 27 \times 13$ 3D rectangle centered at the reference patch. The aim of this experiment is to determine the impact of motion compensation on the proposed method. The results shown correspond to the method VNLB-H with a patch size of $7 \times 7 \times 2$. Some sequences are mainly static, or have a slow motion (*salesman*, *mobile*, *tennis*). They show almost no difference in PSNR when no optical flow is used. The sequences *garden* and *bicycle* have more motion, yet the drop in PSNR caused by assuming zero motion is small, between 0.1dB and 0.3dB. As expected, the sequence which is most affected is *stefan*, which has a very fast camera motion. In this case the difference is quite important: $\approx$ 0.8dB. Nevertheless the result obtained with no motion compensation still outperforms V-BM4D. This shows that the proposed method does not have a critical dependence on the quality of the optical flow, and in fact can still be competitive without any motion estimation.

5.2 Results on color videos

We considered five color test sequences, *tennis*, *coastguard*, *foreman*, *bus* and *football*. The first four were chosen because they were used in [39]. The fifth one, *football*, was added to test the performance of our method on a sequence with complex motions.

The results are shown in Table 5. As for grayscale videos, we compared our method with V-BM4D using the two parameter profiles: V-BM4D-tip and V-BM4D-mp. We also include results of CIFIC [17], a recent method. The proposed method, achieves the highest PSNR values on all the considered sequences and levels of noise, with a significant difference. The difference increases for higher levels of noise. A possible reason for this is that both CIFIC and V-BM4D rely on motion estimation by 2D block matching, which might not be reliable for high noise levels.

To compare with SPTWO, we show in the right half of Table 3 our results on four 8 frame sequences used by [8] (see footnote 8). SPTWO achieves excellent results for the sequence *army* possible due to the fact that its optical flow can be well estimated. On the other sequences, the proposed methods attain a higher PSNR.

Figure 9 shows results obtained by our method and V-BM4D for the sequence *bus*, for noise of $\sigma = 10, 40$. As for grayscale sequences, it can be noticed that some details are better preserved by the proposed method. V-BM4D also shows some color artifacts for high noise values (noticeable for instance on the street *bus*).

6 Conclusions

We presented a Bayesian video denoising algorithm assuming that similar spatio-temporal patches are IID samples from an *a priori* distribution. The proposed method uses motion estimation only

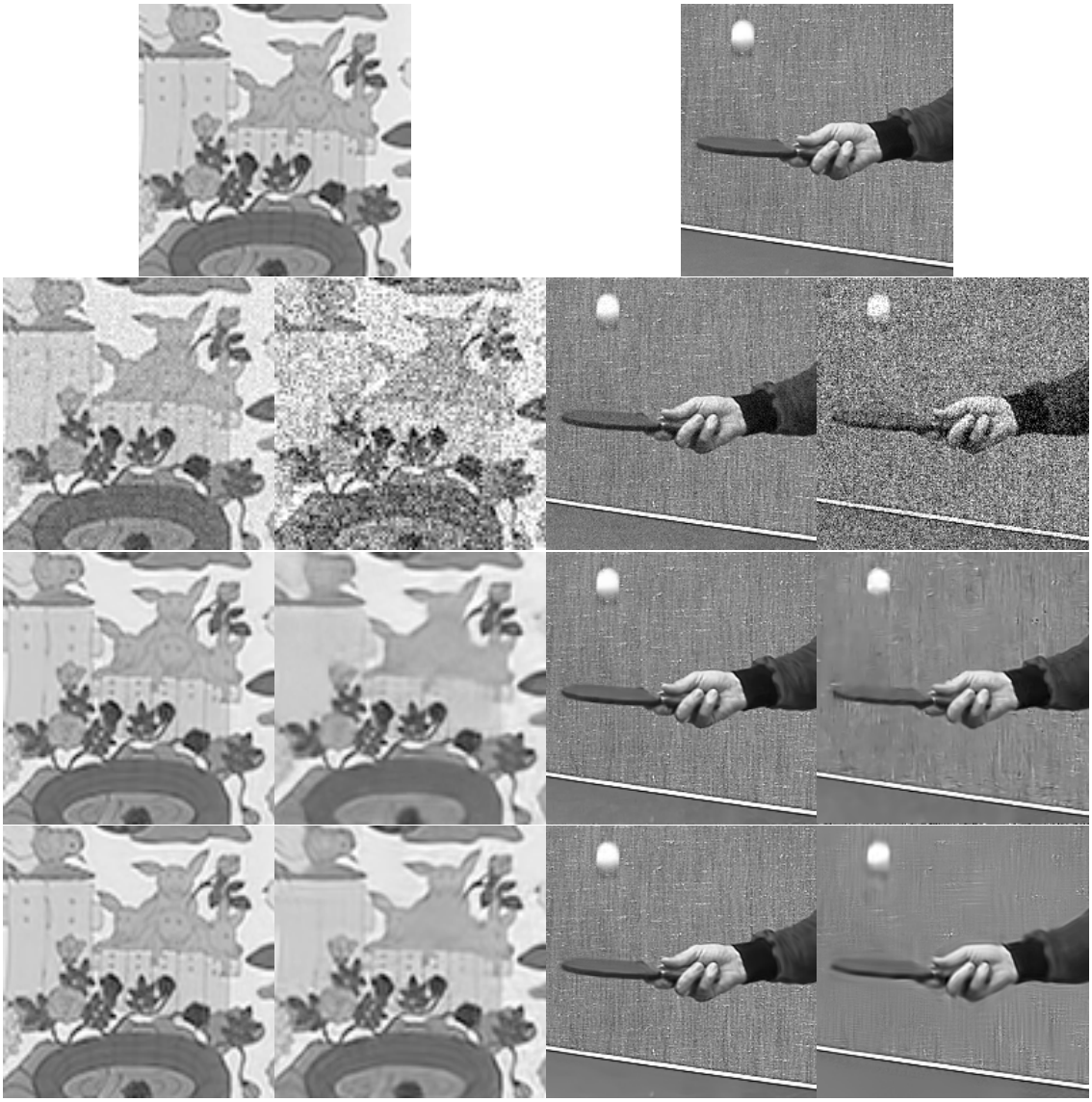

Fig. 7: Details of the results obtained for the grayscale versions of *mobile* and *tennis*. For each sequence, the left column corresponds to $\sigma = 10$ and the right one to $\sigma = 40$. At the top we show the original frame, and the second row shows the noisy frames. In the third row we show the result of V-BM4D-mp for *mobile* and SPTWO for *tennis*. On the bottom row we show our result (VNLB-H $10^2 \times 2$).

to guide the search for similar patches, thus it does not rely on an accurate motion field. It provides state-of-the-art results, both in terms of PSNR and qualitatively.

The patches in each group are estimated using an empirical Wiener filter, which requires estimates of the mean and the covariance matrix of the *a priori* distribution. We studied this estimation problem in the light of the asymptotic results of [18], assuming a low-rank Gaussian prior. This led us to propose two estimators for the eigenvalues of the *a priori* covariance matrix.

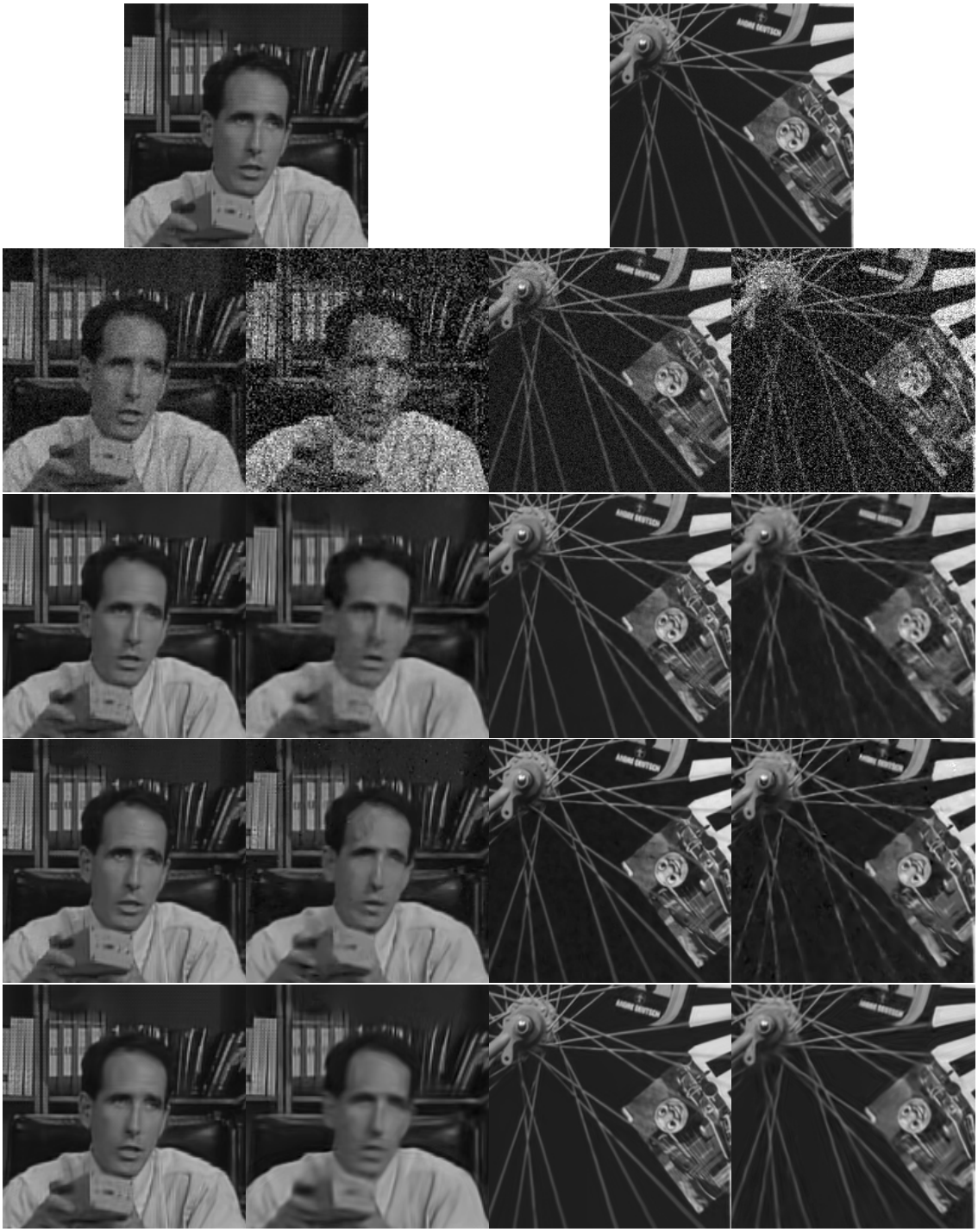

Fig. 8: Details of the results obtained for the grayscale versions of *bicycle* and *salesman*. For each sequence, the left column corresponds to $\sigma = 10$ and the right one to $\sigma = 40$. From top to bottom we show the (1) original frame, (2) noisy frames, (3) the result of V-BM4D-mp, (4) the result of SPTWO and (5) our result (VNLB-H $10^2 \times 2$).

Fig. 9: Details of the results obtained for the color version of *bus*. The left column corresponds to $\sigma = 10$ and the right one to $\sigma = 40$. From top to bottom we show the (1) original frame, (2) noisy frames, (3) the result of V-BM4D-mp, and (4) our result (VNLB-S $10^2 \times 2$).

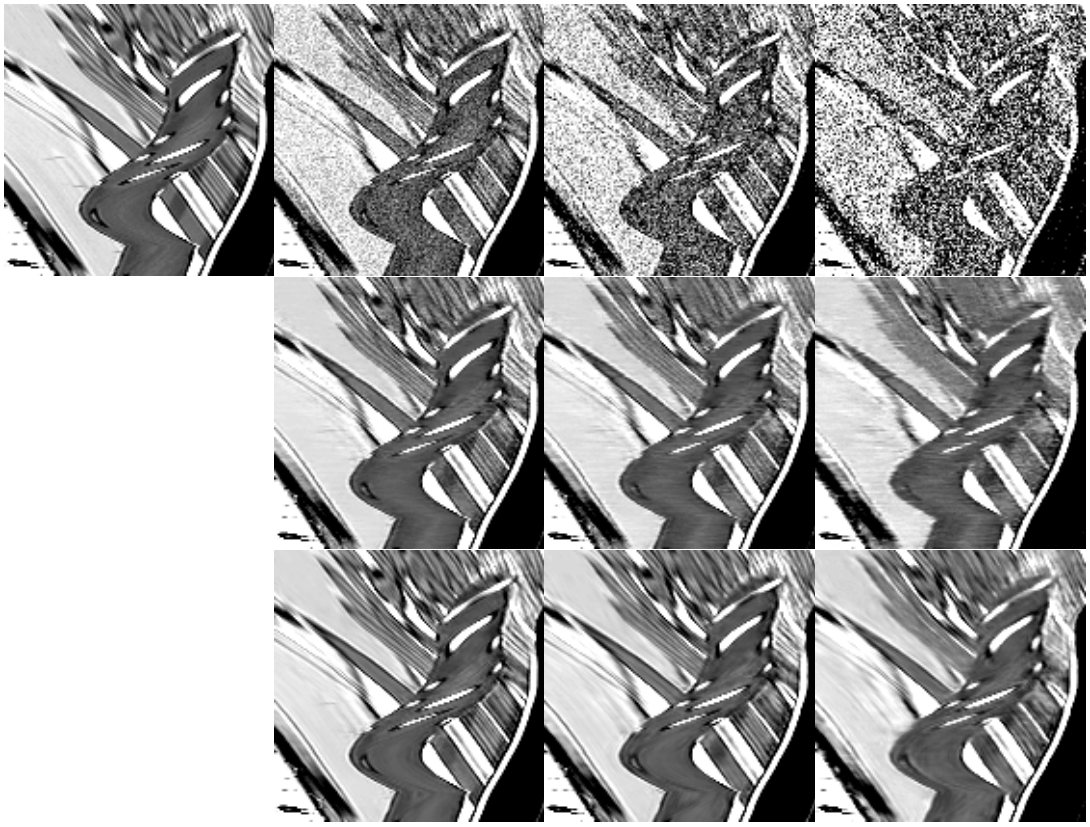

Fig. 10: Visualizing the temporal coherency of the denoising. The images show a horizontal slice of the *mobile* video, at row 220, showing the ball (gray tube) as it gets hit by the toy train (black region in the left). The vertical axis is time, and the horizontal axis is the x axis in the video. On the first row: the original sequence and noisy version with noise 10, 20 and 40. The following rows show the results obtained with V-BM4D, and the results for VNLB-H with patch size $10 \times 10 \times 2$. The result of VNLB varies smoothly in time, reducing the flickering effect. The contrast of the images has been enhanced for better visualization.

One of them is derived by inverting the asymptotic bias of the eigenvalues of the sample covarianace matrix predicted by [18]. This results in a soft-thresholding of such eigenvalues. The resulting estimator performs well when the asymptotic regimes applies, but degrades as the number of patches is reduced. This led us to propose a second estimator consisting on a hard threshold, where the threshold value is controlled by a hand-tuned parameter. Two variants of the video denoising algorithm were proposed: VNLB-S and VNLB-H, based respectively on the soft and hard estimators. VNLB-S is considerably easier to tune, since it has one less parameter per step, and achieves results which are only slightly inferior to those of VNLB-H.

The significant improvement over the state-of-the-art suggests that the Gaussian (or PCA) models that have been successfully applied to still images processing, can also contribute to video restoration. The high-dimensionality of spatio-temporal patches poses challenges in the estimation of the moments of the prior distribution. In this setting, better Bayesian estimators can be obtained

by a theoretically motivated optimization of the empirical Wiener filter, taking into account the uncertainty on the sample covariance matrix.

Another conclusion is that very good results can be obtained without motion estimation. We do not believe that video processing/restoration should be performed in general without taking motion into account, but it is interesting to see how much can be done without compensating motion.

The proposed approach could be extended to tackle more general inverse problems by considering an hyper-Bayesian framework as in [1]. The main obstacle is the computation time. Another logical extension would contemplate including motion deblurring in the Bayesian formulation.

References

1. Aguerrebere, C., Almansa, A., Delon, J., Gousseau, Y., Musé, P.: Inverse Problems in Imaging: a Hyper-prior Bayesian Approach (2016). URL `https://hal.archives-ouvertes.fr/hal-01107519`. Working paper or preprint
2. Arias, P., Morel, J.M.: Towards a Bayesian video denoising method. In: S. Battiato, J. Blanc-Talon, G. Gallo, W. Philips, D. Popescu, P. Scheunders (eds.) Advanced Concepts for Intelligent Vision Systems, *Lecture Notes in Computer Science*, vol. 9386, pp. 107–117. Springer International Publishing (2015). DOI 10.1007/978-3-319-25903-1_10. URL `http://dx.doi.org/10.1007/978-3-319-25903-1_10`
3. Boulanger, J., Kervrann, C., Bouthemy, P.: Space-time adaptation for patch-based image sequence restoration. IEEE Transactions on Pattern Analysis and Machine Intelligence **29**(6), 1096–1102 (2007). DOI 10.1109/TPAMI.2007.1064
4. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, New York, NY, USA (2004)
5. Brailean, J.C., Kleihorst, R.P., Efstratiadis, S., Katsaggelos, A.K., Lagendijk, R.: Noise Reduction Filters for Dynamic Image Sequences: A Review. Proceedings of the IEEE **83**(9), 1272–1292 (1995)
6. Buades, A., Coll, B., Morel, J.M.: Denoising image sequences does not require motion estimation. In: Proceedings of the IEEE Conference on Advanced Video and Signal Based Surveillance, pp. 70–74 (2005)
7. Buades, A., Coll, B., Morel, J.M.: A review of image denoising algorithms , with a new one. Multiscale Modeling and Simulation **4**(2), 490–530 (2006)
8. Buades, A., Lisani, J.L., Miladinović, M.: Patch-based video denoising with optical flow estimation. IEEE Transactions on Image Processing **25**(6), 2573–2586 (2016). DOI 10.1109/TIP.2016.2551639
9. Cai, T., Ma, Z., Wu, Y.: Optimal estimation and rank detection for sparse spiked covariance matrices. Probability Theory and Related Fields **161**(3), 781–815 (2015). DOI 10.1007/s00440-014-0562-z. URL `http://dx.doi.org/10.1007/s00440-014-0562-z`
10. Cai, T.T., Ren, Z., Zhou, H.H.: Estimating structured high-dimensional covariance and precision matrices: Optimal rates and adaptive estimation. Electron. J. Statist. **10**(1), 1–59 (2016). DOI 10.1214/15-EJS1081. URL `http://dx.doi.org/10.1214/15-EJS1081`
11. Chatterjee, P., Milanfar, P.: Patch-based near-optimal image denoising. IEEE Transactions on Image Processing **21**(4), 1635–1649 (2012). DOI 10.1109/TIP.2011.2172799
12. Colom, M., Buades, A.: Analysis and Extension of the Percentile Method, Estimating a Noise Curve from a Single Image. Image Processing On Line **3**, 332–359 (2013). DOI 10.5201/ipol.2013.90
13. Dabov, K., Foi, A.: Image denoising with block-matching and 3D filtering. Electronic ... **6064**, 1–12 (2006). DOI 10.1117/12.643267. URL `http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=728740`
14. Dabov, K., Foi, A., Egiazarian, K.: Video denoising by sparse 3D transform-domain collaborative filtering. In: EUSIPCO, pp. 145–149 (2007). DOI 10.1109/TIP.2007.901238
15. Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.: Image denoising by sparse 3d transform-domain collaborative filtering. IEEE Trans. on IP **16**(8), 2080–2095 (2007)
16. Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.: BM3D Image Denoising with Shape-Adaptive Principal Component Analysis. In: R. Gribonval (ed.) SPARS'09 - Signal Processing with Adaptive Sparse Structured Representations. Inria Rennes - Bretagne Atlantique, Saint Malo, France (2009). URL `https://hal.inria.fr/inria-00369582`
17. Dai, J., Au, O., Pang, C., Zou, F.: Color video denoising based on combined interframe and intercolor prediction. IEEE Transactions on Circuits and Systems for Video Technology **23**(1), 128–141 (2013). DOI 10.1109/TCSVT.2012.2203203
18. Debashis, P.: Asymptotics of sample eigenstructure for a large dimensional spiked covariance model. Statistica Sinica **17**, 1617–1642 (2007)

19. Deledalle, C.A., Salmon, J., Dalalyan, A.: Image denoising with patch based PCA: local versus global. In: Procedings of the British Machine Vision Conference 2011, pp. 25.1–25.10 (2011). DOI 10.5244/C.25.25. URL http://www.bmva.org/bmvc/2011/proceedings/paper25/index.html
20. Elad, M., Aharon, M.: Image denoising via sparse and redundant representations over learned dictionaries. IEEE Transactions on Image Processing **15**(12), 3736–3745 (2006)
21. Ercole, C., Foi, A., Katkovnik, V., Egiazarian, K.: Spatio-temporal pointwise adaptive denoising of video: 3D non-parametric regression approach. In: Proceedings of the First Int. Workshop of Video Processing and Quality Metrics for Consumer Electronics (2005)
22. Ghael, S., Sayeed, A., R.G., B.: Improved wavelet denoising via empirical wiener filtering. In: SPIE Technical Conference on Wavelet Applications in Signal Processing (1997)
23. Ghoniem, M., Chahir, Y., Elmoataz, A.: Nonlocal video denoising, simplification and inpainting using discrete regularization on graphs. Signal Processing **90**(8), 2445–2455 (2010). DOI 10.1016/j.sigpro.2009.09.004. URL http://linkinghub.elsevier.com/retrieve/pii/S016516840900379X
24. Gu, S., Zhang, L., Zuo, W., Feng, X.: Weighted nuclear norm minimization with application to image denoising. The Twenty-Seventh IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2014)
25. Guillemot, T., Almansa, A., Boubekeur, T.: Covariance trees for 2d and 3d processing. The Twenty-Seventh IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2014)
26. Halko, N., Martinsson, P.G., Tropp, J.A.: Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. SIAM Review **53**(2), 217–288 (2011). DOI 10.1137/090771806. URL http://dx.doi.org/10.1137/090771806
27. Jin, F., Fieguth, P., Winger, L.: Wavelet video denoising with regularized multiresolution motion estimation. Eurasip Journal on App. Sig. Proc. **2006**, 1–11 (2006). DOI 10.1155/ASP/2006/72705
28. Johnstone, I.M., Lu, A.Y., Nadler, B., Witten, D.M., Hastie, T., Tibshirani, R., Ramsay, J.O., Johnstone, I.M., Lu, A.Y.: On consistency and sparsity for principal components analysis in high dimensions [with comments]. Journal of the American Statistical Association **104**(486), 682–703 (2009). URL http://www.jstor.org/stable/40592215
29. Kambhatla, N., Leen, T.K.: Dimension reduction by local principal component analysis. Neural Computation **9**, 1493–1516 (1997)
30. Kervrann, C., Boulanger, J.: Optimal spatial adaptation for patch-based image denoising. Image Processing, IEEE Transactions on **15**(10), 2866–2878 (2006). DOI 10.1109/TIP.2006.877529
31. Laus, F., Nikolova, M., Persch, J., Steidl, G.: A nonlocal denoising algorithm for manifold-valued images using second order statistics (2016). URL http://arxiv.org/abs/1607.08481. Preprint
32. Lebrun, M., Buades, A., Morel, J.M.: A Nonlocal Bayesian Image Denoising Algorithm. SIAM Journal on Imaging Sciences **6**(3), 1665–1688 (2013)
33. Lebrun, M., Buades, A., Morel, J.M.: Implementation of the "Non-Local Bayes" (NL-Bayes) Image Denoising Algorithm. Image Processing On Line **3**, 1–42 (2013). DOI 10.5201/ipol.2013.16
34. Lebrun, M., Colom, M., Buades, A., Morel, J.M.: Secrets of image denoising cuisine. Acta Numerica **21**(1), 475–576 (2012)
35. Lebrun, M., Colom, M., Morel, J.M.: The Noise Clinic: a Blind Image Denoising Algorithm. Image Processing On Line **5**, 1–54 (2015). DOI 10.5201/ipol.2015.125
36. Li, W., Zhang, J., Dai, Q.H.: Video denoising using shape-adaptive sparse representation over similar spatio-temporal patches. Signal Processing: Image Communication **26**, 250–265 (2011)
37. Liu, C., Freeman, W.T.: A high-quality video denoising algorithm based on reliable motion estimation. In: ECCV, pp. 706–719 (2010)
38. Maggioni, M., Boracchi, G., Foi, A., Egiazarian, K.: Video Denoising Using Separable 4D Nonlocal Spatiotemporal Transforms. In: Proc. of SPIE, 7870 (2011)
39. Maggioni, M., Boracchi, G., Foi, A., Egiazarian, K.: Video denoising, deblocking, and enhancement through separable 4-D nonlocal spatiotemporal transforms. IEEE Transactions on Image Processing **21**(9), 3952–3966 (2012). DOI 10.1109/TIP.2012.2199324
40. Maggioni, M., Foi, A.: Implementation of V-BM4D. http://www.cs.tut.fi/~foi/GCF-BM3D/
41. Maggioni, M., Katkovnok, V., Egiazarian, K., Foi, A.: A nonlocal transform-domain filter for volumetric data denoising and reconstruction. IEEE Transactions on Image Processing **22**(1), 119–133 (2013)
42. Mairal, J., Sapiro, G., Elad, M.: Learning multiscale sparse representations for image and video restoration. Multiscale Modeling & Simulation **7**(1), 214–241 (2008). DOI 10.1137/070697653. URL http://dx.doi.org/10.1137/070697653
43. Monzón, N., Salgado, A., Sánchez, J.: Robust Discontinuity Preserving Optical Flow Methods. Image Processing On Line **6**, 165–182 (2016). DOI 10.5201/ipol.2016.172
44. Muresan, D.D., Parks, T.W.: Adaptive principal components and image denoising. In: Proceedings 2003 International Conference on Image Processing, vol. 1, pp. I–101–4 vol.1 (2003). DOI 10.1109/ICIP.2003.1246908
45. Nadler, B.: Finite sample approximation results for principal component analysis: A matrix perturbation approach. Ann. Statist. **36**(6), 2791–2817 (2008). DOI 10.1214/08-AOS618. URL http://dx.doi.org/10.1214/08-AOS618

46. Niknejad, M., Rabbani, H., Babaie-Zadeh, M.: Image restoration using Gaussian Mixture Models with spatially constrained patch clustering. Image Processing, IEEE Transactions on **24**(11), 3624–3636 (2015)
47. Oktem, R., Ponomarenko, N.: Image filtering based on discrete cosine transform. Telecommunications and Radio Engineering **66**(18) (2007)
48. Portilla, J., Strela, V., Wainwright, M., Simoncelli, E.: Image denoising using scale mixtures of gaussians in the wavelet domain. Image Processing, IEEE Transactions on **12**(11), 1338–1351 (2003). DOI 10.1109/TIP.2003.818640
49. Protter, M., Elad, M.: Sparse and Redundant Representations and Motion-Estimation-Free Algorithm for Video Denoising. In: Proc. SPIE, 67011, pp. 67,011D–67,011D–12 (2007). DOI 10.1117/12.731851
50. Protter, M., Elad, M.: Image sequence denoising via sparse and redundant representations. IEEE Transactions on Image Processing **18**(1), 27–35 (2009)
51. Rajpoot, N., Yao, Z., Wilson, R.: Adaptive wavelet restoration of noisy video sequences. In: Image Processing, 2004. ICIP '04. 2004 International Conference on, vol. 2, pp. 957–960 Vol.2 (2004). DOI 10.1109/ICIP.2004.1419459
52. Salmon, J., Harmany, Z., Deledalle, C.A., Willett, R.: Poisson noise reduction with non-local pca. Journal of Mathematical Imaging and Vision **48**(2), 279–294 (2014). DOI 10.1007/s10851-013-0435-6. URL `http://dx.doi.org/10.1007/s10851-013-0435-6`
53. Sánchez, J., Meinhardt-Llopis, E., Facciolo, G.: TV-L1 Optical Flow Estimation. Image Processing On Line **3**, 137–150 (2013). DOI 10.5201/ipol.2013.26
54. Selesnick, I.W., Li, K.Y.: Video denoising using 2D and 3D dual-tree complex wavelet transforms. Proceedings of SPIE - The International Society for Optical Engineering **5207**(2), 607–618 (2003). DOI 10.1117/12.504896
55. Seshadrinathan, K., Bovik, A.C.: Motion tuned spatio-temporal quality assessment of natural videos. Trans. Img. Proc. **19**(2), 335–350 (2010). DOI 10.1109/TIP.2009.2034992. URL `http://dx.doi.org/10.1109/TIP.2009.2034992`
56. Sutour, C., Deledalle, C.A., Aujol, J.F.: Adaptive regularization of the nl-means: Application to image and video denoising. IEEE Transactions on Image Processing **23**(8), 3506–3521 (2014). DOI 10.1109/TIP.2014.2329448
57. Tipping, M.E., Bishop, C.M.: Mixtures of probabilistic principal component analysis. Neural Computation **11**, 443–482 (1999)
58. Yu, G., Sapiro, G.: DCT image denoising: a simple and effective image denoising algorithm. Image Processing On Line **1** (2011). DOI 10.5201/ipol.2011.ys-dct
59. Yu, G., Sapiro, G., Mallat, S.: Solving inverse problems with piecewise linear estimators: From gaussian mixture models to structured sparsity. Image Processing, IEEE Transactions on **21**(5), 2481–2499 (2012). DOI 10.1109/TIP.2011.2176743
60. Zach, C., Pock, T., Bischof, H.: A Duality Based Approach for Realtime TV-L1 Optical Flow, pp. 214–223. Springer Berlin Heidelberg, Berlin, Heidelberg (2007). DOI 10.1007/978-3-540-74936-3_22. URL `http://dx.doi.org/10.1007/978-3-540-74936-3_22`
61. Zhang, L., Dong, W., Zhang, D., Shi, G.: Two-stage image denoising by principal component analysis with local pixel grouping. Pattern Recognition **43**(4), 1531 – 1549 (2010)
62. Zlokolica, V., Ptžurica, A., Philips, W.: Wavelet-domain video denoising based on reliability measures. IEEE Transactions on Circuits and Systems for Video Technology **16**(8), 993–1007 (2006). DOI 10.1109/TCSVT.2006.879994
63. Zoran, D., Weiss, Y.: From learning models of natural image patches to whole image restoration. In: Computer Vision (ICCV), 2011 IEEE International Conference on, pp. 479–486 (2011). DOI 10.1109/ICCV.2011.6126278