Video denoising by combining patch search and CNNs

Axel Davy, Thibaud Ehret, Jean-Michel Morel, Pablo Arias, Gabriele Facciolo

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

Video denoising by combining patch search and CNNs

Axel Davy · Thibaud Ehret · Jean-Michel Morel · Pablo Arias · Gabriele Facciolo

Received: date / Accepted: date

Abstract Non-local patch based methods were until recently the state of the art for image denoising but are now outperformed by CNNs. In video denoising however, they are still competitive with CNNs, as they can effectively exploit the video temporal redundancy, which is a key factor to attain high denoising performance. The problem is that CNN architectures are not compatible with the search for self-similarities. In this work we propose a simple, yet efficient way to feed video self-similarities to a CNN. The non-locality is incorporated into the network via a first non-trainable layer which finds for each patch in the input image its most similar patches in a search region. The central values of these patches are then gathered in a feature vector which is assigned to each image pixel. This information is presented to a CNN which is trained to predict the clean image. We apply the proposed method to image and video denoising. In the case of video, the patches are searched for in a 3D spatio-temporal volume. The proposed method achieves state-of-the-art results.

Keywords Denoising · Video denoising · Non-local · Patch-based methods · CNN

The authors gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan V GPU used for this research. Work partly financed by IDEX Paris-Saclay IDI 2016, ANR-11-IDEX-0003-02, ONR grant N00014-17-1-2552, CNES MISS project, DGA Astrid ANR-17-ASTR-0013-01, DGA ANR-16-DEFA-0004-01, MENRT. This work used HPC resources from the "Mésocentre" computing center of CentraleSup 'elec and ENS Paris-Saclay supported by CNRS and Région Île-de-France.

All authors were with Centre Borelli, ENS Paris-Saclay, CNRS, Université Paris-Saclay, 91190 Gif-sur-Yvette, France
E-mail: axel.davy@ens-paris-saclay.fr

1 Introduction

Advances in image sensor hardware have steadily improved the acquisition quality of image and video cameras. However, a low signal-to-noise ratio is unavoidable in low lighting conditions if the exposure time is limited (for example to avoid motion blur). This results in high levels of noise, which negatively affects the visual quality of the video and hinders its use for many applications. As a consequence, denoising is a crucial component of any camera pipeline. Furthermore, by interpreting denoising algorithms as proximal operators, several inverse problems in image processing can be solved by iteratively applying a denoising algorithm [52]. Hence the need for video denoising algorithms with a low running time.

Literature review on image denoising. Image denoising has a vast literature where a variety of methods have been applied: PDEs and variational methods (including MRF models) [12, 54, 55], transform domain methods [23], non-local (or patch-based) methods [7, 19], multiscale approaches [28], etc. See [36] for a review. In the last two or three years, CNNs have taken over the state of the art. In addition to attaining better results, CNNs are amenable to efficient parallelization on GPUs potentially enabling real-time performance. We can distinguish two types of CNN approaches to image denoising: *trainable inference networks* and *black box* networks.

In the first type, the architecture mimics the operations performed by a few iterations of optimization algorithms used for MAP inference with MRFs prior models. Some approaches are based on the Fields-of-Experts model [54], such as [5, 15, 58]. The architecture of [66] is based on EPLL [74], which models the

a priori distribution of image patches as a Gaussian mixture model. Trainable inference networks reflect the operations of an optimization algorithm, which leads in some cases to unusual architectures, and to some restrictions in the network design. For example, in the *trainable nonlinear reaction diffusion network* (TNRD) of [15] even layers must be an image (i.e. have only one feature). As pointed out in [34] these architectures have strong similarities with the residual networks of [30].

The black-box approaches treat denoising as a standard regression problem, not using much of the domain knowledge acquired during decades of research in denoising. The first denoising approaches using neural networks were proposed in the mid and late 2000s. Jain and Seung [33] proposed a five layer CNN with 5×5 filters, with 24 features in the hidden layers and sigmoid activation functions. Burger et al. [11] reported the first results competitive with patch-based methods using a multilayer perceptron trained to denoise 17×17 patches, but with a heavy architecture. More recently, DnCNN [71] obtained impressive results with a far lighter 17 layer deep CNN with 3×3 convolutions, ReLU activations and batch normalization [32]. This work also proposes a blind denoising network that can denoise an image with an unknown noise level $\sigma \in [0, 55]$, and a multi-noise network trained to denoise blindly three types of noise. A faster version of DnCNN, named FFDNet, was proposed in [72], which also allows handling spatially varying noise by adding a noise map $\sigma(x)$ as an additional input. The architectures of DnCNN and FFDNet keep the same image size throughout the network. Other architectures [13, 44, 57] use pooling or strided convolutions to downscale the image, and then up-convolutional layers to upscale it back. Skip connections connect the layers before the pooling with the output of the up-convolution to avoid loss of spatial resolution. Skip connections are used extensively in [61].

Although these architectures produce very good results, for textures formed by repetitive patterns non-local patch-based methods still perform better [11, 71]. Some works have therefore attempted to incorporate the non-local patch similarity into a CNN framework. Qiao *et al.* [50] proposed inference networks derived from the non-local FoE MRF model [60]. This can be seen as a non-local version of the TNRD network of [15]. A different non-local TNRD was introduced by [37]. BM3D-net [69] pre-computes for each pixel a stack of similar patches and feeds them into a CNN that reproduces the operations done by (the first step of) the BM3D algorithm: a linear transformation of the group of patches, a non-linear shrinkage function and a second linear transform (the inverse of the first). The linear transformations and the shrinkage function are the trainable parameters. In [17] the authors propose an iterative approach that can be used to reinforce non-locality to any denoiser. Each iteration consists of the application of the denoiser followed by a non-local filtering step. An inconvenience is that the resulting algorithm requires to iterate the denoising network. Trainable non-local modules have been recently proposed by using differentiable relaxations of the nearest neighbor [39] and k nearest neighbors [48] selection rules, or using Graph CNNs [65], where the graph edges encode the non-local connections between pixels with similar features. These approaches are very interesting as they allow the combination of local and non-local interactions in a trainable way.

Literature review on video denoising. CNNs have been successfully applied to several video processing tasks such as deblurring [59], video frame synthesis [41] or super-resolution [31, 56], but their application to video denoising has been limited so far. In [14] a recurrent architecture is proposed, but the results are below the state of the art. More recently, Tassano *et al.* [62] proposed DVDnet, a convolutional architecture which processes five consecutive frames to predict the central frame. Each frame is first denoised spatially, and then warped to frame t via an optical flow. The aligned frames are stacked together with the central frame and processed by a "temporal denoising" network. The authors use a non-trainable optical flow, which prevents the network from being trained end-to-end. Two recent works proposed networks without explicit motion estimation: ViDeNN-G [16] processes three consecutive frames, and applies first a spatial denoising followed by temporal denoising, similar to [62], except that the frames are stacked without aligning them. A different architecture, named fastDVDnet, was proposed in [63]. Instead of first using a spatial denoising, three consecutive noisy frames are stacked together. The stack is processed by a U-net [53] which predicts the central frame. To extend the temporal receptive field of the network, the authors cascade two levels of these networks. The overall network takes five frames as input. Recently [25, 29, 46] focused on the related problem of image burst denoising reporting very good results. There is also recent work focusing on unknown noise-model denoising for videos that use self-supervision for training [25, 26].

Before the advent of CNNs, patch-based methods were the state of the art [3, 10, 18, 24, 43, 67], and some continue to be competitive in terms of denoising performance [3, 24], albeit with a larger computational cost). They exploit extensively the self-similarity of natural

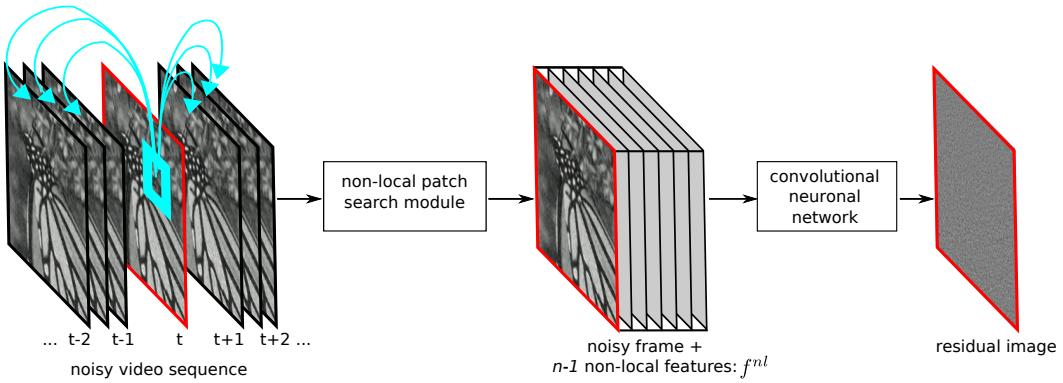

Fig. 1: Illustration of the proposed *Video Non-Local Network* (VNLnet). The first module performs a patch-wise nearest neighbor search across neighboring frames. Then, the current frame, and the feature vectors f^{nl} of each pixel (the center pixels of the nearest neighbors) are fed into the network.

images and videos, namely the fact that most patches have several similar patches around them (spatially and temporally). Each patch is denoised using these similar patches, which are searched for in a region around it. The search region generally is a space-time cube, but more sophisticated search strategies have also been used. Because of the use of such broad search neighborhoods these methods are called *non-local*. While these video denoising algorithms perform very well, they often are computationally costly.

Patch-based methods usually follow three steps that can be iterated: (1) search for similar patches, (2) denoise the group of similar patches, (3) aggregate the denoised patches to form the denoised frame. VBM3D [18] improves the image denoising algorithm BM3D [19] by searching for similar patches in neighboring frames using a "predictive search" strategy which speeds up the search and gives some temporal consistency. VBM4D [43] generalizes this idea to 3D patches. In VNLB [2], video extension of [35], spatio-temporal patches that were not motion compensated are used to improve the temporal consistency. In [24] a generic search method extends every patch-based denoising algorithm into a global video denoising algorithm by extending the patch search to the entire video. SPTWO [10] and DDVD [9] use optical flow to warp the neighboring frames to each target frame. Each patch of the target frame is then denoised using the similar patches in this volume with a Bayesian strategy similar to [35]. Recently, [67] proposed to learn an adaptive optimal transform using batches of frames.

Patch-based approaches have also been applied in frame-recursive denoising methods [4, 27], that denoise each frame using only the corresponding noisy frame and the previous denoised frame.

Contributions. In this work we propose a strategy to exploit non-local information with a CNN in the context of image and video denoising. It works particularly well in the case of video denoising, where it achieves state-of-the-art results.

The method first computes for each image patch the n most similar neighbors in a rectangular spatio-temporal search window and gathers their central pixels forming a feature vector which is assigned to each image location. This results in an image with n channels, which is fed to a CNN trained to predict the clean image from this high dimensional vector. We trained our network for grayscale and color video denoising. Practically, training this architecture is made possible by a GPU implementation of the patch search that allows computing the nearest neighbors efficiently.

To train our network we created a dataset of 17k video segments. In the two testing datasets, our network obtains state-of-the-art results on both color and grayscale video denoising. The code to generate the datasets and reproduce our results is available online[1]. A preliminary version of this work was presented in [21]. The present version includes an extension to color videos, a detailed comparison with recent works, extended discussions comparing these methods, and new experiments.

2 Proposed method

Let u be a grayscale video and $u(x,t)$ denote its value at pixel position x in frame t. We observe v, a noisy version of u contaminated by additive white Gaussian noise:

$$v = u + r, \quad \text{where, } r(x,t) \sim \mathcal{N}(0,\sigma^2). \tag{1}$$

[1] The code to reproduce our results, the training and testing datasets can be found at `https://github.com/axeldavy/vnlnet`.

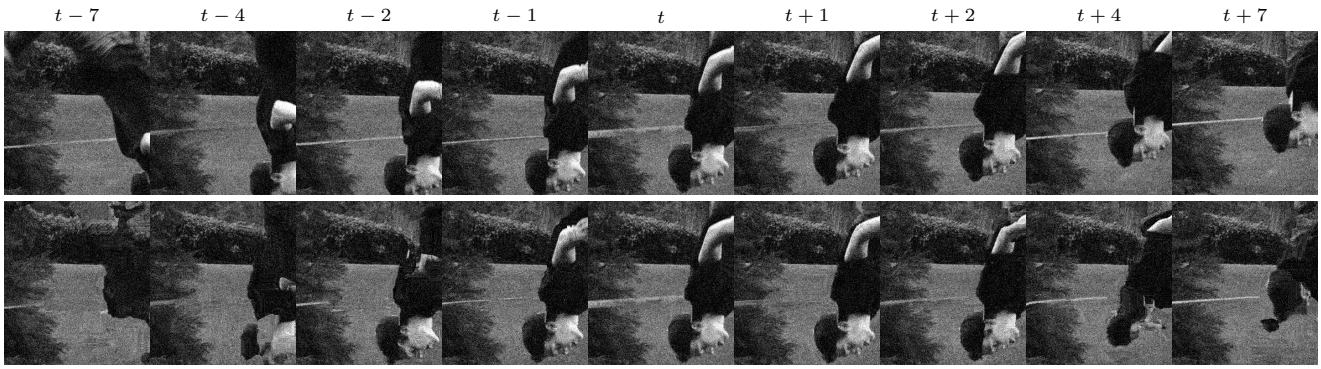

Fig. 2: Spatio-temporal video crop (top row) and the features extracted by the non-local search for the central frame (bottom row), applying the restriction of one neighbor per frame.

Our video denoising network processes the video frame by frame. Before it is fed to the network, each frame is pre-processed by a non-local patch search module which computes a non-local feature vector at each image position. A diagram of the proposed method is shown in Figure 1. We call our method VNLnet for *Video Non-Local Network*.

2.1 Non-local features

Each frame in the video is pre-processed by the non-local patch search module to produce a 3D tensor f^{nl} of n channels. This is the input to the network. The parameter n is the number of nearest neighbor patches searched for each pixel by this pre-processing module.

Let P be a patch of size $s \times s$ centered at pixel x in frame t. The patches are arranged as vectors with s^2 components. The patch search module computes the L_2 distances between the patch P and the patches in a 3D rectangular search region of size $w_s \times w_s \times w_t$ centered at (x, t). The positions of the n most similar patches are (x_i, t_i), ordered either by increasing distance or by increasing frame index t_i.

The pixel values at those positions are gathered to produce the n-dimensional non-local feature vector associated to pixel (x, t):

$$f^{\text{nl}}(x,t) = [v(x_1,t_1), ..., v(x_n,t_n)]. \quad (2)$$

The non-local features correspond to n images which resemble the frame $v(\cdot, t)$ (see Figure 2). One of them is $v(\cdot, t)$ itself, as the reference patch P is always among the n nearest neighbors. The remaining $n-1$ images are built from the central pixels of the most similar patches to each patch in $v(\cdot, t)$.

One neighbor per frame. We will also consider a restricted version of the patch search, not allowing more than one match per-frame. In this variant, we set the number of matches n to be equal to the number of frames in the search region, resulting in a match for each frame. The neighbors are sorted by frame index instead of the patch distance. With this configuration, the ith non-local feature map corresponds to a warped version of the ith neighboring frame, aligned to the reference frame t. An example is shown in Figure 2, for $n = 15$. This can be related to [62, 68], which align the input frames using an optical flow. Indeed patch matching can be seen as a rough optical flow with integer displacements.

Other alternatives. The proposed non-local features are inspired by classical patch-based methods such as [8]. Selecting similar patches with the L_2 distance and extracting their central pixel is possibly the simplest way of providing the network with a non-local context. The L_2 distance is widely used, and was shown to be optimal for additive white Gaussian noise (AWGN) [22]. The same work also shows that other patch distances should be used for other noise distributions.

We considered some alternatives to feeding only the central pixels to the network: such as providing as well the patch similarities (which could be used by the network to ignore bad matches), or extracting a small patch (3×3 or 5×5) around the center of every matching patch, instead of the central pixel. However, we did not observe a significant improvement to justify the increase in complexity: the differences in the validation PSNR for these variants were within a 0.1dB range. The reason for this is probably that the added pixels in the patch around the central pixel are redundant with the central pixels of the similar patches for neighboring pixels.

In a previous version of our work [21] we also included four 1×1 convolutional layers as a trainable transformation of the non-local features. We removed

them in this work as we found that the same performance can be achieved with a simpler design.

2.2 Network architecture

For processing the non-local features, we considered standard architectures used in image restoration modifying their input layer to the n channel input tensor. We have tested three such architectures.

DnCNN. The DnCNN architecture was proposed in [71] for still image denoising. It consists of a feed-forward network with 17 layers with 64 3×3 convolution kernels, each one followed by batch normalization and ReLU activations. The output layer is a 3×3 convolution. The network outputs a residual image, which has to be subtracted to the noisy image to get the denoised one.

U-Net. U-Nets (also called hourglass networks) have been introduced in [53] for image segmentation. This architecture consists of an encoder-decoder convolutional network. The encoder part downscales the input by a series of pooling layers or strided convolutions, whereas the decoder upscales the coarse features to the desired output resolution. The differentiating aspect of U-Nets from previous encoder-decoder networks are skip connections that bypass each coarser scale. These networks have been applied to many tasks, among them denoising of both single images [13, 57] and videos [62]. In our experiments, we use the U-Nets blocks of [63], except that the first two layers, which feature group convolutions and a number of kernels computed from the number of input frames, are replaced by two standard convolution layers of 32 kernels.

EDSR. EDSR [38] is a residual CNN architecture tuned for super-resolution. The main architectural differences with DnCNN are the introduction of skip connections every two convolution layers, and the removal of the batchnorm layers. Thus in our experiments we took our DnCNN networks and applied these two changes (the number of convolutional layers and their parameters are kept).

3 Datasets and training

We denote by $\mathcal{F}$ the application of the network. The input to $\mathcal{F}$ at time t is $f_t^{\mathrm{nl}} = f^{\mathrm{nl}}(\cdot, t)$, the n-channel image with non-local features gathered from a window of frames around t. The training loss is the mean square error (MSE) between the reconstructed frame and the ground truth clean frame:

$$l(\mathcal{F}(f_t^{\mathrm{nl}}), u_t) = ||\mathcal{F}(f_t^{\mathrm{nl}}) - u_t||_2^2. \tag{3}$$

We use residual training, as in [71]. This means that the network actually predicts the noise, and therefore the denoised image is obtained by subtracting the predicted noise from the noisy input.

For RGB videos, we compute the patch search on grayscale frame resulting from averaging the color channels. To form the non-local features we take the RGB components of the central pixels of the matching patches, resulting in a input tensor with $3n$ channels. For RGB videos we only trained a DnCNN network, with 25 layers and 96 channels (i.e. about three times the number of parameters used for grayscale video).

3.1 Datasets

For the training and validation sets we used a database of short segments of 16 frames extracted from YouTube videos. Only HD videos with Creative Commons license were used. From each video we extracted several segments, separated by at least $10s$. In total the database consists of 16950 segments extracted from 1068 videos, organized in 64 categories (such as antelope, cars, factory, etc.). As the original videos might contain compression artifacts, noise, etc, we downscaled the video to a height of 540 pixels. This removes the minor artifacts of the videos and better represents clean targets. In addition, we randomized the anti-aliasing filter width (Gaussian blur) of the downscaling. This results in a variety of sharpness/blur in the training dataset, and thus helps reducing dataset bias. We separated 6% of the videos of the database for the validation (one video for each category). For grayscale networks, grayscale data is obtained by converting the previous color datasets.

For training we ignored the first and last frames of each segment for which the 3D patch search window did not fit in the video. During validation we only considered the central frame of each sequence. The resulting validation score is thus computed on 503 sequences (1 frame each).

For testing, we used two distinct datasets. The first one is the dataset of [1], which was extracted from the Derf's Test Media collection.[2] It is composed of seven sequences of 100 frames of size 960×540. In this dataset, the camera is either still or has a smooth motion. The second one is the `test-dev` split of the DAVIS video

[2] `https://media.xiph.org/video/derf`

segmentation challenge [49]. It consists of 30 videos having between 25 and 90 frames. In this dataset, the motion is more challenging. In order to remove compression artifacts and noise present in the original images, both datasets were obtained with a similar downscaling as for the training set (the original images ranged between HD and 4K). Each dataset was processed using a different anti-aliasing filter width.

3.2 Training details

At each training epoch, first a subset of the videos of the dataset is selected and noise is added to generate noisy samples. Second the non-local patch search module is run on every video selected. This results in *videos of non-local features* where each frame has n channels containing the output of the patch search module. Third the network is trained on mini-batches built from small crops extracted at random positions on the videos of non-local features.

During training, we ignore spatio-temporal border effects by excluding the first and last $w_t/2$ frames and ignoring crops at borders. At testing time, we simply extended the video by mirroring it at the start and the end of the sequence and adding black borders for the patch-search module.

The training epochs comprise 17000 mini-batches of size 64 square crops of 44 pixels width. We trained for 30 epochs using a combination of recent new optimization techniques which give small performance improvements on various machine learning tasks: Lookahead [73], RAdam [40] and Gradient Centralization [70].[3] In addition, we reduced the learning rate at epochs 15 and 27 (from $1e^{-3}$ to $1e^{-4}$ and $1e^{-6}$ respectively). Training a network takes about 14 hours on a NVIDIA TITAN V for grayscale videos, and 24 hours for color videos.

4 Experiments and parameter tuning

In this section we evaluate the effect of the non-local features first in still image denoising, and then, after studying the impact of the parameters, in video denoising. Unless otherwise stated, the results reported were obtained using a DnCNN architecture for the fusion network. The network is trained from scratch for each different parameter setting.

[3] We used the implementation of `https://github.com/lessw2020/Ranger-Deep-Learning-Optimizer`

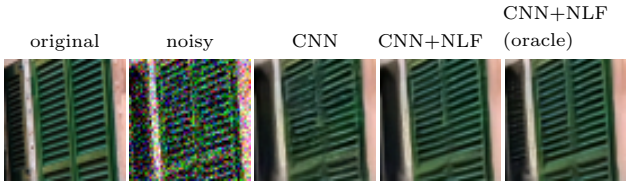

Fig. 3: Results on still image denoising (AWGN with $\sigma = 25$). Original clean image, noisy image, result obtained with the baseline CNN, result of incorporating the non-local features by finding the nearest neighbors on the noisy image or the oracle noise-less image. Contrast has been linearly scaled for better visualization.

4.1 The potential of non-locality

Although the focus of this work is in video denoising, it is still interesting to study the performance of the proposed non-local CNN on images. Figure 3 shows a comparison between a baseline CNN (a 15 layer DnCNN [71]) and a version of our method trained for still image denoising (it collects 9 neighbors by comparing 9×9 patches and uses a 15 layer DnCNN architecture. The non-local features are sorted by patch distance. The results with and without non-local information are very similar. The only differences are visible on very self-similar parts like the shutters in Figure 3. This is confirmed by quantitative results. The average PSNR on the CBSD68 dataset [45, 71] (noise with $\sigma = 25$) obtained for the baseline CNN is of 31.24dB. The non-local CNN only leads to a 0.04dB improvement (31.28dB). The figure also shows the result of an oracular method: the nearest neighbor search is performed on the noise-free image, though the pixel values are taken from the noisy image. The method obtains an average PSNR of 31.85dB, 0.6dB over the baseline. The oracular results show that non-locality has a great potential to improve the results of CNNs. However, this improvement is hindered by the difficulty of finding accurate matches in the presence of noise. A way to reduce the matching errors is to use larger patches. But on images, larger patches have fewer matches. In contrast, as we will see below, the temporal redundancy of videos allows using very large patches.

4.2 Parameter tuning for video denoising

The non-local search has three main parameters: The patch size, the number of retained matches and the size of in the search region (both spatially and in number of frames). We expect the best matches to be past or future versions of the current patch, so we set the number of matches as the number of frames on which we search.

Patch size	no patch	9×9	15×15	21×21	31×31	41×41
ONPF	33.88	35.70	36.62	37.07	37.39	37.51
free	33.88	35.68	36.55	36.95	37.20	37.32

(a) Impact of the patch size (with 15 frames)

# search frames	no patch	3	7	11	15
ONPF	33.88	35.85	36.93	37.27	37.51

(b) Impact of number of frames (patch size is 41 × 41)

search size	5 × 5	11 × 11	21 × 21	41 × 41	51 × 51
ONPF	36.84	37.13	37.35	37.51	37.53

(c) Impact of search region size (patch size is 41 × 41, with 15 frames, ONPF)

Table 1: Effect of patch size, search region size and number of frames (the number of neighbors is kept equal to the number of frames). All values correspond to the PSNR computed on the validation set for noise with $\sigma = 20$. In 1a we compare the performance with the one-neighbor-per-frame restriction (ONPF) and without it (free).

# fusion net	DnCNN	U-Net	EDSR
PSNR	37.51	37.42	37.55

Table 2: Impact of the architecture of the CNN used for fusing the features.

# stacked frames	1	3	7	11	15
PSNR	33.88	35.56	36.22	36.42	36.57

Table 3: Impact of the number of frames considered for the *video-DnCNN* network (the network input is a 3D crop rather than the result of the non-local search). PSNR on the validation set AWGN with $\sigma = 20$.

In the following we study the impact of the different parameters. Note that for each different parameter, we retrain the fusion network to make sure that it is the optimal one for the chosen parameters. In all cases, we consider denoising of grayscale videos with $\sigma = 20$.

Patch size. In Table 1a, we explore the impact of the patch size used for the matching. For each patch size we show results with the unconstrained patch search and with the restriction of one neighbor per frame (ONPF). In both cases, the results improve with the patch size, even for patches much larger than the ones typically used in patch-based methods. The ONPF restriction produces a slight improvement in performance, particularly for larger patches. Figure 4 shows visual results obtained with the ONPF restriction. As the patch size grows, there is a noticeable increase in the amount of details recovered from the background.

Number of frames in the search region. In Table 1b and Figure 5, we see the impact of the number of frames used in the search window (and thus the number of nearest neighbors). One can see that the more frames, the better. Increasing the number of frames beyond 15 (7 past, current, and 7 future) does not justify the small increase of performance. Foreground moving objects are unlikely to get good neighbors for the selected patch size, unlike background objects, thus it comes to no surprise that the visual quality of the background improves with the number of patches, while foreground moving objects (for example the legs in Figure 5) do not improve much.

Spatial size of the search region. Results varying the size of the search region are shown in Table 1c. The results improve for larger search regions, although with diminishing returns. From 41×41 to 51×51 there is no significant improvement. Our search region is not compensated by motion, thus having a large search region is necessary to find matching patches on distant frames for moving objects.

Choice of the fusion network. We have evaluated the performance obtained with the two other architectures for the fusion network described in Section 2. The results obtained are shown in Table 2. The three networks were trained with the same training set and the same procedure. All networks achieve a very similar PSNR, within a 0.13dB range. This shows that the fusion architecture is not critical.

Throughout the rest of the paper, we shall use 41×41 patches and a search volume of 41×41 pixels and 15 frames as the parameters for the patch search. For the fusion network we will use the DnCNN architecture.

4.3 Discussion

4.3.1 Surprisingly large patches

An immediate question is why do such big patches achieve such a good performance. On one hand, increasing the patch size has the effect of reducing the variance of the noise in the patch distance, resulting in matches that are less affected by it. This explains the details recovered in the background in Figures 4 and 5. The large size allow to reliably match the background patches even if the texture seems to be completely obfuscated by noise.

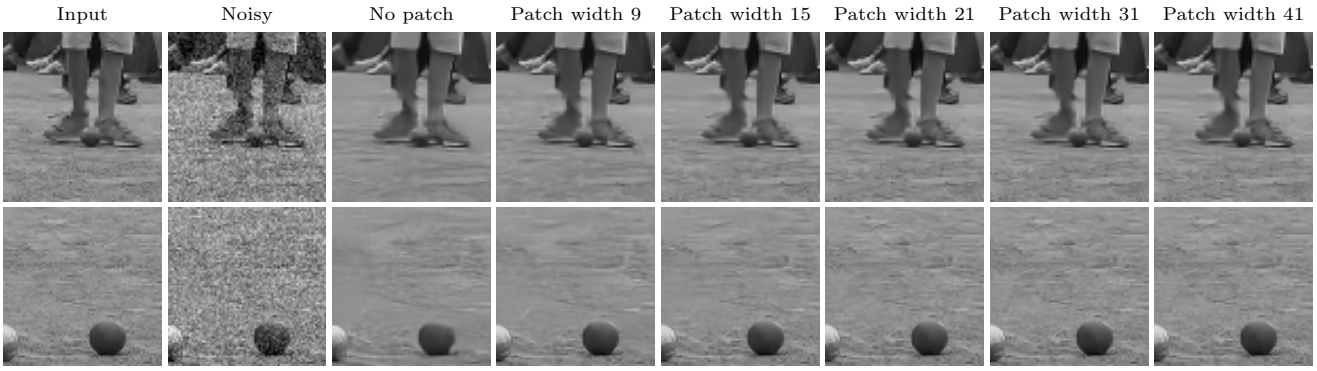

Fig. 4: Example of denoised results with our method when changing the patch size, respectively no patch search, 9×9, 15×15, 21×21, 31×31 and 41×41 patches. The 3D search window has 15 frames for these experiments.

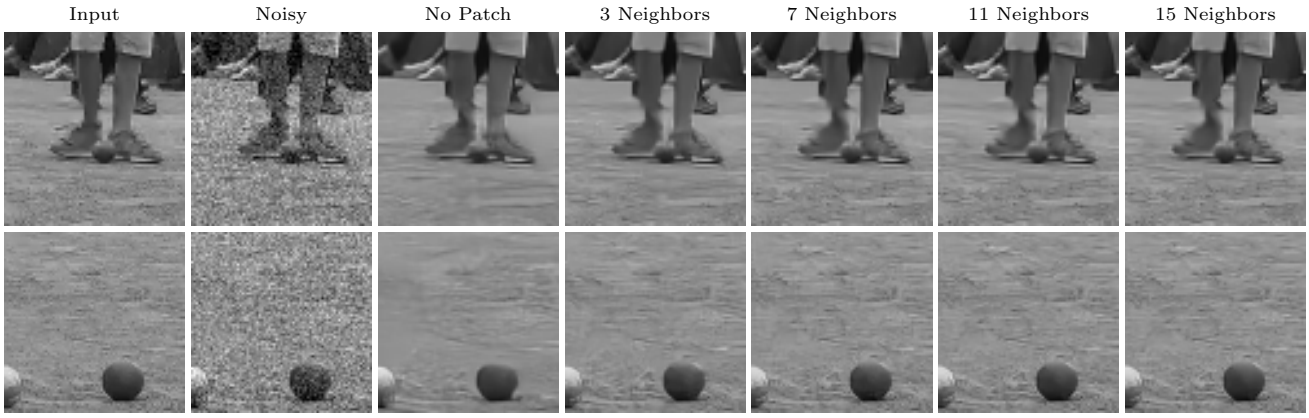

Fig. 5: Example of denoised results with our method when changing the number of frames considered in the 3D search window (respectively no patch search, 3, 7, 11 and 15). 41×41 patches were used for these experiments.

On the other hand, a large patch size becomes a disadvantage for objects with a non-translational motion. In these cases, finding similar matches in the neighboring frames becomes less likely for larger patches. This can be seen in Figure 2. The person in the foreground is rotating as she performs a backflip jump. Patches in the foreground object have similar matches only for the closest neighboring frames (e.g. the face cannot be reconstructed for $t \pm 4$), whereas patches in the background can be matched throughout all the temporal extend of the search region. As a consequence, different pixels will have a different number of relevant non-local features. The fusion network learns during training to be robust to these bad matches. It seems to be able to identify when the patch search fails, relying only on the non-local features that are correlated with the target pixel, and adapting accordingly the amounts of spatial denoising and non-local denoising.

This phenomenon can be observed in Figure 6, where we compare the results of the proposed VNLnet against the single frame DnCNN from [71] and a *Non-Local Pixel Mean* (NLPM), which simply averages the values of the non-local features given by the patch-matching (please ignore for the moment the column labeled *video-DnCNN*). On the first two rows, the motion is consistent on the whole crop, and thus the output of NLPM is sharp, indicating good matches. As a result, VNLnet's output shows more details than the single frame DnCNN. However the Non-Local Pixel Mean output is blurry for the person in the third row and the background of the fourth row. For these regions, VNLnet and DnCNN both have similar quality. Meanwhile, for the regions of these two crops with good matches (the background of the third row, and the person of the fourth row), the quality is improved. Using bigger patches will increase the number of patches covering regions of conflicting motion. As a result, we see that the performance gain from 31×31 to 41×41 is rather small.

4.3.2 A note on motion handling

To evaluate the effectiveness of the non-local features we also train a network with the same architecture, but

Fig. 6: Example of denoised result for *Non-Local Pixel Mean*, DnCNN, video-DnCNN (see Section 4.3.2) and VNLnet, for AWGN with $\sigma = 20$. The four crops highlight the results on frames feature various kinds of motion. The videos are part of the DAVIS dataset. *Non-Local Pixel Mean* corresponds to the average of the output of the non-local patch search.

instead of feeding it with the n non-local features we feed it with the stack of the n neighboring frames. We call this network video-DnCNN. We do this for different values of n, and show the results obtained on Table 3.

The performance of the video-DnCNN network increases with the number of frames, although less than the proposed VNLnet (see Table 1b). In particular, for video-DnCNN the average PSNR on the validation set stagnates at 11 frames. The reason for this is that while the denoising performance increases on sequences with majority of small and smooth motions, it drops significantly when there are many large or irregular motions.

Without the non-local patch-search module, the network has to learn to handle motion implicitly, which makes the task significantly harder. As the number of input frames increases, so does the complexity of the internal motion compensation the network has to learn to denoise accurately. The video-DnCNN network then overfits to the most frequent motion patterns in the training set, and fails when it encounters a different motion. By factoring out the motion, the non-local patch search module removes the need for the network to learn to adapt to various types of motion, enabling a better generalization on various moving scenes.

This can be seen in Figure 6, by comparing the results of video-DnCNN and VNLnet, for objects with fast/irregular motion patterns. VNLnet is able to recover much more details, thanks to the patch search.

Figure 7 shows the PSNR gain of VNLnet over video-DnCNN for each sequence on the grayscale DAVIS test set. The gain given by the non-local patch-search module is significant, except only for two sequences. These feature fixed cameras and static backgrounds covering most of the frame. The sequences with larger gains have complex motion.

Texture is also an important aspect, as patches with a distinct and highly contrasted texture can be matched more reliably. In Figure 8 we plot the per patch PSNR

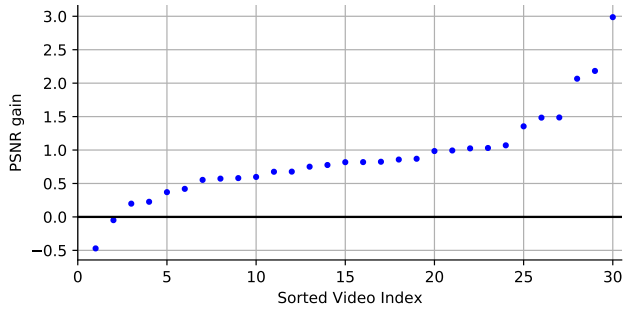

Fig. 7: PSNR gain of VNLnet versus the network without patch search of Table 3 (15 input frames) on grayscale DAVIS ($\sigma = 20$) (35.48db versus 34.59db).

gain of VNLnet against the video-DnCNN method and a single frame DnCNN, in terms of a measure of patch "texturedness". We measure patch texturedness as the sum of the squared gradient magnitudes in a patch:

$$T(x,t) = \sum_h \|\nabla u(x+h,t)\|^2 \tag{4}$$

where h varies in a patch centered at x, ∇u is spatial gradient of $u(\cdot,t)$ and $\|\cdot\|$ is the Euclidean norm. The texturedness is computed on the clean video. We denoised the grayscale DAVIS test set with $\sigma = 20$, and for each 41×41 patch, we compute its texturedness $T(x,t)$ and the PSNR gain, which we denote by $G(x,t)$. We bin these quantities in a 2D histogram with 400×400 bins which is illustrated by the grayscale image in Figure 8 (darker pixels correspond to bins with higher frequency). The figure also shows in red the mean PSNR gain for each level of texturedness, and in blue the same mean, but multiplied by the relative frequency of each texturedness level in the dataset.

The figures confirm that the PSNR gain with respect to both video-DnCNN and single image DnCNN increases with patch texturedness. As expected, the gain is larger with respect to the single frame DnCNN. For patches with very low texturedness the video-DnCNN performs better than the VNLnet. This makes sense, as for those patches the matching will be influenced by the noise. A similar behaviour has been observed for patch-based methods in global video denoising [24] and external denoising [47]. Such patches are relatively rare in the test set, which explains the overall positive gains in Figure 7.

A related question is whether better results could be obtained replacing the patch search by a frame matching using a standard optical flow. The results of Table 1a highlight the importance of very reliable matches, and thus the optical flow would have to be chosen with care. In [63], the authors of DVDnet [62] stressed the

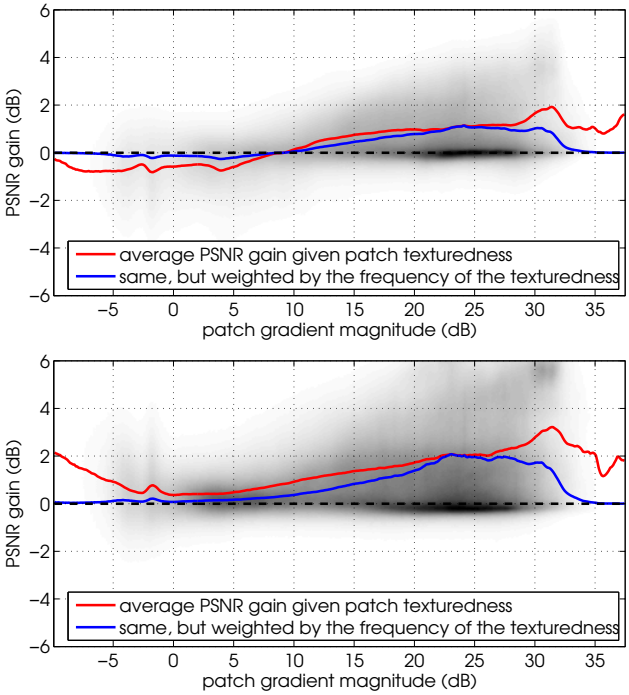

Fig. 8: Patch PSNR gain between VNLnet versus video-DnCNN (top) and DnCNN (bottom) on grayscale DAVIS with $\sigma = 20$, as a function of a measure of the patch *texturedness* (defined here by the patch gradient magnitude). The grayscale image depicts the 2D histogram of PSNR gains and patch texturedness computed from all 41×41 patches in the test set. The red curve shows the average PSNR gain for each texturedness level, and the blue curve shows the same average PSNR gain, but weighted by the frequency of each texturedness level.

difficulty of finding a fast and reliable optical flow, and moved away from it for FastDVDnet [63]. In [68], the optical flow is computed with a reduced version of SpyNet [51], which is trained together with the rest of the network. In our case, the patch search module is not trainable, and the network is trained to process its output.

5 Comparison with other methods

In this section, we compare the proposed method VNLnet to VBM3D [18], VNLB [2], and DnCNN [71] for grayscale videos, and VBM3D [18], VNLB [2], DnCNN [71], ViDeNN-G [16], DVDnet [62], and FastDVDnet [63] for color videos. DnCNN was applied frame-by-frame.

We trained grayscale and color networks for AWGN of standard deviation 10, 20 and 40. To highlight the fact that a CNN method can be easily re-targeted to

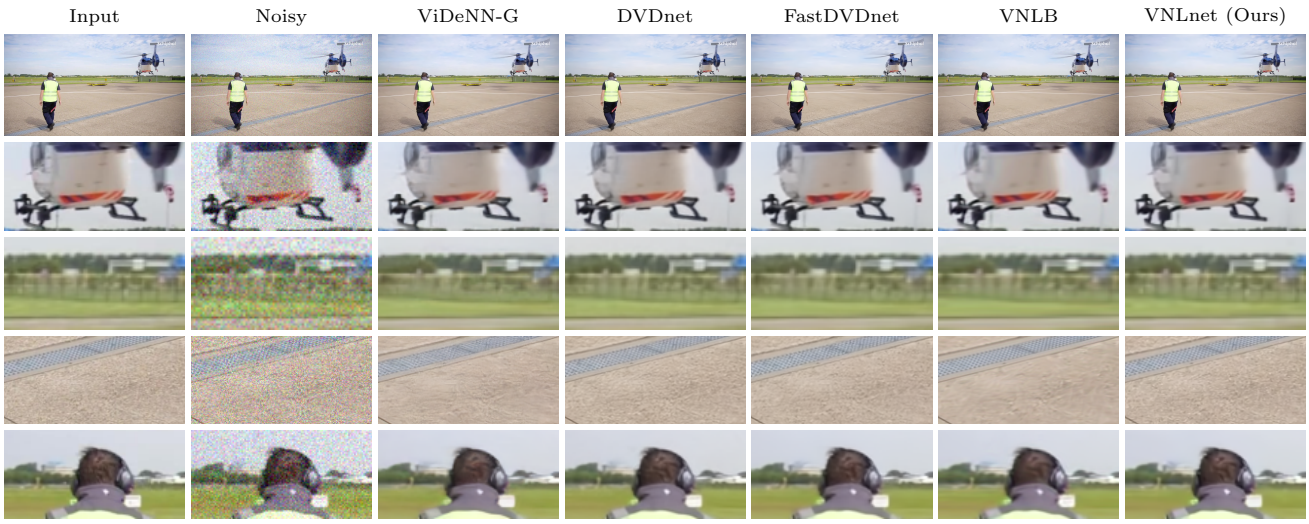

Fig. 9: Example of denoised result for several algorithms (AWGN with $\sigma = 20$) on a sequence of the color DAVIS dataset [49]. The crops highlight the results on non-moving and moving parts of the video.

		Method	$\sigma = 10$	$\sigma = 20$	$\sigma = 40$
GRAYSCALE	DERF	SPTWO	39.56 / .9675	35.99 / .9368	30.93 / .7901
		VBM3D	38.24 / .9599	34.68 / .9100	31.11 / .8360
		VBM4D	38.88 / .9534	35.10 / .9169	31.40 / .8432
		VNLB	**40.57 / .9731**	**36.81 / .9428**	**32.95 / .8856**
		DnCNN	37.28 / .9482	33.60 / .8973	30.09 / .8156
		VNLnet	***40.22 / .9730***	***36.51 / .9415***	***32.60 / .8772***
			Corr. Gaussian noise		Uniform S&P 25%
		VNLB	***25.39 / .5922***		***23.49 / .7264***
		VNLnet	**32.92 / .8899**		**48.05 / .9952**
	DAVIS		$\sigma = 10$	$\sigma = 20$	$\sigma = 40$
		VBM3D	37.43 / .9425	33.75 / .8870	30.12 / .8068
		VNLB	***38.84 / .9634***	***35.26 / .9240***	***31.88 / .8622***
		DnCNN	36.80 / .9451	32.94 / .8878	28.69 / .7940
		VNLnet	**39.10 / .9661**	**35.53 / .9305**	**32.03 / .8692**
COLOR	DERF	VBM3D	38.19 / .9560	34.80 / .9165	31.65 / .8568
		VNLB	**40.93 / .9760**	**37.62 / *.9528***	**33.97** / .9042
		DnCNN	38.00 / .9588	34.44 / .9171	31.14 / .8520
		ViDeNN-G	38.16 / .9588	35.34 / .9291	32.25 / .8757
		DVDnet	39.08 / .9689	36.48 / .9474	33.43 / ***.9051***
		FastDVDnet	39.01 / .9669	36.16 / .9427	33.21 / .9010
		VNLnet	***40.46 / .9748***	***37.36* / .9542**	***33.79* / .9079**
	DAVIS	VBM3D	38.43 / .9591	34.74 / .9157	31.38 / .8473
		VNLB	***40.31 / .9725***	***36.79*** / .9420	33.34 / .8896
		DnCNN	38.91 / .9655	35.24 / .9278	31.81 / .8637
		ViDeNN-G	38.46 / .9619	35.47 / .9314	32.32 / .8756
		DVDnet	39.31 / .9702	36.66 / ***.9488***	***33.61* / .9059**
		FastDVDnet	39.74 / .9714	36.50 / .9457	33.35 / .9013
		VNLnet	**40.70 / .9760**	**37.32 / .9528**	**33.72 / *.9054***

Table 4: Quantitative comparison (PSNR and SSIM) of other methods versus the proposed VNLnet on two test sets, both in grayscale and in color. We highlighted the best performance in bold and the second best in bold italics.

different noise distributions, we also trained a grayscale network for Gaussian noise correlated by a 3×3 box kernel such that the final standard deviation is $\sigma = 20$, and 25% uniform Salt and Pepper noise (removed pixels are replaced by random uniform noise).

Table 4 shows the denoising results obtained on the two compared datasets. For grayscale videos, we also include results for SPTWO [10] and VBM4D [42], computed in [1] for the DERF dataset. Figures 9 and 10 show results for the most relevant RGB methods. VNLB (Video Non-Local Bayes) outperformed on average all other methods on the DERF dataset. Meanwhile on the DAVIS dataset, our method performed the best both in grayscale and color, for all the three tested noise levels. VNLB ranked second, except in color for high noise levels, where it was surpassed by DVDnet and FastDVDnet.

A comparison of the results for grayscale and color in Table 4 reveals that CNN-based methods exploit better the correlations between color channels: while for grayscale, VBM3D significantly outperforms DnCNN in PSNR on the DAVIS dataset, the reverse occurs for color. Moreover, VNLnet performed proportionally better in color than in grayscale. This should not come as a surprise, since the way in which VBM3D and VNLB treat color is rather heuristic: an orthogonal color transform is applied to the video which is supposed to decorrelate color information. Then the processing of each color channel of a group of patches is done independently.

In order to better compare qualitative aspects of the results we show some details in Figures 9 and 10. For some scenes, VNLnet recovers significantly more details in the background, as shown in Figure 9. In general, we observe that VNLnet, and the other video CNN methods (ViDeNN-G, DVDnet, and FastDVDnet) have better background reconstruction than VNLB. This can be seen in Figure 9 and Figure 10. Some details however

Fig. 10: Examples of areas where the level of restored detail of the methods differs significantly (AWGN with $\sigma = 40$) in videos of DERF.

are better recovered by VNLB and VNLnet. For example in Figure 10 both methods recover the red lights in the top left corner of the image in the first column, while for the three other methods the lights do not appear. In the second column, VNLB does not reconstruct the trees as well as the CNN-based methods, but manages to recover the color of the clothes of the person in the bottom left. VNLnet not only recovers better the tree structure, but also recovers the clothes correctly. None of the methods restore satisfyingly the grass texture in the third column of Figure 10 for the tested noise level. This highlights that there is room for improvement. All five methods achieve reasonable temporal consistency, which is an important quality requirement for video denoising.

One of the benefits of CNNs over traditional model-based approaches is that they can be easily targeted to handle other noise distributions by simply re-training them. We demonstrate this by reporting the results of our method on salt-and-pepper noise and correlated noise in Table 4. We include the result of VNLB as a reference.

In summary, the proposed approach for video denoising obtains state-of-the-art results on both test sets. In particular, it outperforms previous CNN approaches. In light of this, we can conclude that effectively exploiting a large number of surrounding frames is key. Indeed, the proposed VNLnet uses 15 frames, in comparison to the 3 frames used by ViDeNN-G and 5 frames of DVDnet and FastDVDnet. Most of these methods avoid relying on an explicit optical flow computation, which can be unreliable given that the input frames are noisy. FastDVDnet and ViDeNN-G do so by performing an early fusion of triplets of consecutive frames without alignment. The non-local features computed via patch correspondences result in an effective way to present the information of large frame neighborhoods to a network that merges them. Training the fusion network is then straightforward.

5.1 Running times and number of parameters

In Tables 5 and 6, we compare the running time in grayscale and color of several methods when denoising a video frame. As before, we consider DnCNN as the fusion network. In Table 5, the compared methods are run on a single CPU core, while in Table 6, a system with an Intel Xeon W-2145 and a NVIDIA TITAN V is used. For both tables, the loading and writing time of the videos were subtracted. Since we do not have a CPU implementation of the patch search layer, we cannot measure a CPU time for VNLnet. On the GPU, for grayscale videos of 960×540 the non-local search takes 822ms, where as the DnCNN fusion network takes 95ms. Extrapolating this, we could expect a CPU time 8 to 9 times slower than DnCNN.

Most of the running time of our method is spent in the patch search. Our GPU implementation of patch search is similar to the convolution-based patch search described in [20]. With the default parameters, the non-local search is costly because matches are searched in 15 frames for patches centered at every pixel of our image. These parameters can be modified, trading off speed by denoising quality. In the Tables 7 we show how the time spent in the patch search as a function of the patch size, spatial search region size and number of frames. These behave as expected: the time grows linearly with the number of frames, and quadratically with respect to the patch and search sizes. For example, with a search region of 21×21, the per frame patch search time is reduced by more than three times, while the drop in PSNR is only 0.16dB (see Table 1c). The table also shows the patch search time normalized by the number of frames that need to be aligned in the search region.

The implementation could be further accelerated by reducing the size of the 3D window using tricks explored in other papers. VBM3D for example centers the search on each frame on small windows around the best matches found in the previous frame. A related acceleration is to use a search strategy based on PatchMatch [6].

One of the benefits of the proposed approach is that the fusion network can be simple, resulting in an easier training and a smaller memory footprint of the network weights. The three tested architectures were designed for single image denoising. This reduces the number of trainable parameters, in comparison with other video processing networks. The DnCNN and EDSR networks have both around 0.56M parameters, while the U-Net has 1M. For RGB videos, our DnCNN with 25 layers and 96 channels has 1.9M parameters, roughly the same as, FastDVDnet [63], which requires 2M parameters as it uses two U-Nets.

Other works such as [62, 68] use an optical flow subnetwork for aligning the frames. This has the advantage that the alignment can be trained together with the fusion network [68]. However, the smallest current optical flow networks have between 1M and 5M parameters [64], and take around 50ms to 100ms (depending on the desired quality) to estimate the optical flow between a pair of frames.

6 Conclusions

We described an effective way of incorporating temporal non-local information into a CNN for video de-

VBM3D	DnCNN	VBM4D	VNLB	SPTWO
1.3s	13s	52s	140s	210s

Table 5: Running time per frame on a grayscale 960 × 540 video for VBM3D, DnCNN, VBM4D, VNLB and SPTWO on a single CPU core.

VNLB	ViDeNN-G	DVDnet	FastDVDnet	VNLnet
26.94s	0.186s	2.67s	0.074s	1.16s

Table 6: Running time per frame on a color 960 × 540 video for VNLB, ViDeNN, DVDnet, FastDVDnet, VNLnet on a system with a 16-cores CPU (Intel Xeon W-2145) and a NVIDIA TITAN V.

	9×9	15×15	21×21	31×31	41×41
Per frame	126	201	327	545	822
Normalized	9	14	23	39	59

(a) Average patch search time per frame (in ms) for different patch sizes on a 960 × 540 video (with a 41 × 41 × 15 search region), and the averaged time normalized by the depth of the search region (minus the central frame).

	3	7	11	15
Per frame	118	346	585	822
Normalized	59	58	59	59

(b) Average patch search time per frame (in ms) for different numbers of frames in the search region on a 960 × 540 video (patch size is 41 × 41, search region spatial size is 41 × 41), and the averaged time normalized by the depth of the search region (minus the central frame).

	5 × 5	11 × 11	21 × 21	41 × 41	51 × 51
Per frame	54	100	253	822	1269
Normalized	4	7	18	59	91

(c) Average patch search time per frame (in ms) for different search region spatial sizes on a 960 × 540 video (patch size is 41 × 41, with 15 frames), and the averaged time normalized by the depth of the search region (minus the central frame).

Table 7: Time spent in the patch search, for different values of the parameters. We report both the average total time spent in the search (considering all frames in the search region) and the time normalized by the depth of the search region (minus the central frame for which there is no search).

noising. The proposed method computes for each image patch the n most similar neighbors on a spatio-temporal window and gathers their central pixels to form a non-local feature vector which is given to a CNN. Our method yields a significant gain compared to other CNN approaches. It has similar performance to the best non-CNN method evaluated, VNLB, outperforming it on the largest of our test datasets. In addition, we noted that CNN approaches tend to better reconstruct backgrounds than VNLB, which are perceptually relevant areas. To prevent dataset bias we also proposed a public training set comprising 17k videos from 64 different categories and a simulation strategy that emulates different levels of sharpness.

We have seen the importance of reliable matches: On the validation set, the best performing method used patches of size 41 × 41 for the patch search. We have also noticed that on regions with non-reliable matches (complex motion), the network seems to revert to a result similar to single image denoising. Thus we believe future works should focus on improving this area.

References

1. Arias, P., Facciolo, G., Morel, J.M.: A comparison of patch-based models in video denoising. In: IEEE Image, Video, and Multidimensional Signal Processing Workshop, pp. 1–5. IEEE (2018)
2. Arias, P., Morel, J.M.: Towards a bayesian video denoising method. In: Advanced Concepts for Intelligent Vision Systems, LNCS. Springer (2015)
3. Arias, P., Morel, J.M.: Video denoising via empirical bayesian estimation of space-time patches. Journal of Mathematical Imaging and Vision **60**(1), 70–93 (2018)
4. Arias, P., Morel, J.M.: Kalman Filtering of Patches for Frame-Recursive Video Denoising. In: The IEEE Conference on Computer Vision and Pattern Recognition Workshops (2019)
5. Barbu, A.: Training an active random field for real-time image denoising. IEEE Transactions on Image Processing **18**(11), 2451–2462 (2009). DOI 10.1109/TIP.2009.2028254
6. Barnes, C., Shechtman, E., Finkelstein, A., Goldman, D.B.: Patchmatch: A randomized correspondence algorithm for structural image editing. In: ACM SIGGRAPH 2009 Papers. Association for Computing Machinery (2009). DOI 10.1145/1576246.1531330
7. Buades, A., Coll, B., Morel, J.M.: Denoising image sequences does not require motion estimation. In: IEEE Conference on Advanced Video and Signal Based Surveillance, pp. 70–74 (2005)
8. Buades, A., Coll, B., Morel, J.M.: A non-local algorithm for image denoising. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 60–65. IEEE (2005)
9. Buades, A., Lisani, J.L.: Dual domain video denoising with optical flow estimation. In: IEEE International Conference on Image Processing, pp. 2986–2990 (2017). DOI 10.1109/ICIP.2017.8296830

10. 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
11. Burger, H.C., Schuler, C.J., Harmeling, S.: Image denoising: Can plain neural networks compete with BM3D? In: 2012 IEEE conference on computer vision and pattern recognition, pp. 2392–2399. IEEE (2012)
12. Caselles, V., Chambolle, A., Cremers, D., Novaga, M., Pock, T.: An introduction to total variation for image analysis. Theoretical Foundations and Numerical Methods for Sparse Recovery, De Gruyter, Radon Series Comp. Appl. Math. **9**, 263–340 (2010)
13. Chen, C., Chen, Q., Xu, J., Koltun, V.: Learning to see in the dark. In: The IEEE Conference on Computer Vision and Pattern Recognition (2018)
14. Chen, X., Song, L., Yang, X.: Deep rnns for video denoising. In: Applications of Digital Image Processing (2016)
15. Chen, Y., Pock, T.: Trainable Nonlinear Reaction Diffusion: A Flexible Framework for Fast and Effective Image Restoration. IEEE Transactions on Pattern Analysis and Machine Intelligence **39**(6), 1256–1272 (2017). DOI 10.1109/TPAMI.2016.2596743
16. Claus, M., van Gemert, J.: Videnn: Deep blind video denoising. In: IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 0–0 (2019)
17. Cruz, C., Foi, A., Katkovnik, V., Egiazarian, K.: Nonlocality-reinforced convolutional neural networks for image denoising. IEEE Signal Processing Letters **25**(8), 1216–1220 (2018). DOI 10.1109/LSP.2018.2850222
18. Dabov, K., Foi, A., Egiazarian, K.: Video denoising by sparse 3d transform-domain collaborative filtering. In: 2007 15th European Signal Processing Conference, pp. 145–149. IEEE (2007)
19. Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.: Image denoising by sparse 3-D transform-domain collaborative filtering. IEEE Transactions on image processing **16**(8), 2080–2095 (2007)
20. Davy, A., Ehret, T.: GPU acceleration of NL-means, BM3D and VBM3D. Journal of Real-Time Image Processing pp. 1–18 (2020)
21. Davy, A., Ehret, T., Morel, J., Arias, P., Facciolo, G.: A non-local CNN for video denoising. In: IEEE International Conference on Image Processing. IEEE (2019)
22. Deledalle, C.A., Denis, L., Tupin, F.: How to compare noisy patches? Patch similarity beyond Gaussian noise. International Journal of Computer Vision **99**(1), 86–102 (2012)
23. Donoho, D.L., Johnstone, J.M.: Ideal spatial adaptation by wavelet shrinkage. Biometrika **81**(3), 425–455 (1994). DOI 10.1093/biomet/81.3.425
24. Ehret, T., Arias, P., Morel, J.M.: Global patch search boosts video denoising. In: International Conference on Computer Vision Theory and Applications (2017)
25. Ehret, T., Davy, A., Arias, P., Facciolo, G.: Joint demosaicing and denoising by overfitting of bursts of raw images. In: IEEE International Conference on Computer vision (2019)
26. Ehret, T., Davy, A., Morel, J.M., Facciolo, G., Arias, P.: Model-blind video denoising via frame-to-frame training. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 11369–11378 (2019)
27. Ehret, T., Morel, J.M., Arias, P.: Non-local Kalman: A recursive video denoising algorithm. In: IEEE International Conference on Image Processing, pp. 3204–3208. IEEE (2018)
28. Facciolo, G., Pierazzo, N., Morel, J.M.: Conservative scale recomposition for multiscale denoising (the devil is in the high frequency detail). SIAM Journal on Imaging Sciences **10**(3), 1603–1626 (2017). DOI 10.1137/17M1111826
29. Godard, C., Matzen, K., Uyttendaele, M.: Deep burst denoising. In: European Conference on Computer Vision, pp. 538–554 (2018)
30. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016). DOI 10.1109/CVPR.2016.90
31. Huang, Y., Wang, W., Wang, L.: Bidirectional recurrent convolutional networks for multi-frame super-resolution. In: Advances in Neural Information Processing Systems 28, pp. 235–243. Curran Associates, Inc. (2015)
32. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. CoRR **abs/1502.03167** (2015)
33. Jain, V., Seung, S.: Natural image denoising with convolutional networks. In: Advances in Neural Information Processing Systems 21, pp. 769–776. Curran Associates, Inc. (2009)
34. Kobler, E., Klatzer, T., Hammernik, K., Pock, T.: Variational networks: Connecting variational methods and deep learning. In: V. Roth, T. Vetter (eds.) Pattern Recognition, pp. 281–293. Springer International Publishing, Cham (2017)
35. Lebrun, M., Buades, A., Morel, J.M.: A nonlocal bayesian image denoising algorithm. SIAM Journal on Imaging Sciences **6**(3), 1665–1688 (2013)

36. Lebrun, M., Colom, M., Buades, A., Morel, J.M.: Secrets of image denoising cuisine. Acta Numerica **21**(1), 475–576 (2012)
37. Lefkimmiatis, S.: Non-local color image denoising with convolutional neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 5882–5891 (2017). DOI 10.1109/CVPR.2017.623
38. Lim, B., Son, S., Kim, H., Nah, S., Mu Lee, K.: Enhanced deep residual networks for single image super-resolution. In: IEEE Conference on computer vision and pattern recognition workshops, pp. 136–144 (2017)
39. Liu, D., Wen, B., Fan, Y., Loy, C.C., Huang, T.S.: Non-local recurrent network for image restoration. In: Advances in Neural Information Processing Systems, pp. 1680–1689 (2018)
40. Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., Han, J.: On the variance of the adaptive learning rate and beyond. CoRR **abs/1908.03265** (2019)
41. Liu, Z., Yeh, R.A., Tang, X., Liu, Y., Agarwala, A.: Video frame synthesis using deep voxel flow. In: IEEE International Conference on Computer Vision, pp. 4463–4471 (2017)
42. Maggioni, M., Boracchi, G., Foi, A., Egiazarian, K.: Video Denoising Using Separable 4D Nonlocal Spatiotemporal Transforms. In: Proc. of SPIE (2011)
43. 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 (2012)
44. Mao, X., Shen, C., Yang, Y.B.: Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In: Advances in Neural Information Processing Systems 29, pp. 2802–2810. Curran Associates, Inc. (2016)
45. Martin, D., Fowlkes, C., Tal, D., Malik, J., et al.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: International Conference on Computer vision, vol. 2, pp. 416–423. IEEE (2001)
46. Mildenhall, B., Barron, J.T., Chen, J., Sharlet, D., Ng, R., Carroll, R.: Burst denoising with kernel prediction networks. In: IEEE Conference on Computer Vision and Pattern Recognition (2018)
47. Mosseri, I., Zontak, M., Irani, M.: Combining the power of internal and external denoising. In: IEEE International Conference on Computational Photography, pp. 1–9 (2013)
48. Plötz, T., Roth, S.: Neural nearest neighbors networks. In: International Conference on Neural Information Processing Systems, NIPS'18, pp. 1095–1106. Curran Associates Inc., USA (2018)
49. Pont-Tuset, J., Perazzi, F., Caelles, S., Arbeláez, P., Sorkine-Hornung, A., Gool, L.V.: The 2017 davis challenge on video object segmentation. CoRR **abs/1704.00675** (2017)
50. Qiao, P., Dou, Y., Feng, W., Li, R., Chen, Y.: Learning non-local image diffusion for image denoising. In: ACM International Conference on Multimedia, MM '17, pp. 1847–1855. ACM, New York, NY, USA (2017). DOI 10.1145/3123266.3123370
51. Ranjan, A., Black, M.J.: Optical flow estimation using a spatial pyramid network. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 4161–4170 (2017)
52. Romano, Y., Elad, M., Milanfar, P.: The little engine that could: Regularization by denoising (RED). SIAM Journal on Imaging Sciences **10**(4), 1804–1844 (2017). DOI 10.1137/16M1102884
53. Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional Networks for Biomedical Image Segmentation. MICCAI pp. 234–241 (2015). DOI 10.1007/978-3-319-24574-4{_}28
54. Roth, Stefan and Black, Michael J: Fields of experts: a framework for learning image priors. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 860–867 vol. 2 (2005). DOI 10.1109/CVPR.2005.160
55. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Phys. D **60**(1-4), 259–268 (1992). DOI 10.1016/0167-2789(92)90242-F
56. Sajjadi, M.S.M., Vemulapalli, R., Brown, M.: Frame-recurrent video super-resolution. In: The IEEE Conference on Computer Vision and Pattern Recognition (2018)
57. Santhanam, V., Morariu, V.I., Davis, L.S.: Generalized deep image to image regression. CoRR **abs/1612.03268** (2016)
58. Schmidt, U., Roth, S.: Shrinkage fields for effective image restoration. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2774–2781 (2014). DOI 10.1109/CVPR.2014.349
59. Su, S., Delbracio, M., Wang, J., Sapiro, G., Heidrich, W., Wang, O.: Deep video deblurring for hand-held cameras. In: IEEE Conference on Computer Vision and Pattern Recognition (2017)
60. Sun, J., Tappen, M.F.: Learning non-local range markov random field for image restoration. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2745–2752 (2011). DOI 10.1109/CVPR.2011.5995520

61. Tai, Y., Yang, J., Liu, X., Xu, C.: Memnet: A persistent memory network for image restoration. IEEE International Conference on Computer Vision pp. 4549–4557 (2017)
62. Tassano, M., Delon, J., Veit, T.: Dvdnet: A fast network for deep video denoising. In: IEEE International Conference on Image Processing (2019)
63. Tassano, M., Delon, J., Veit, T.: Fastdvdnet: Towards real-time deep video denoising without flow estimation. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1354–1363 (2020)
64. Teed, Z., Deng, J.: RAFT: Recurrent all-pairs field transforms for optical flow. In: European Conference on Computer Vision (2020)
65. Valsesia, D., Fracastoro, G., Magli, E.: Image denoising with graph-convolutional neural networks. In: IEEE International Conference on Image Processing, pp. 2399–2403. IEEE (2019)
66. Vemulapalli, R., Tuzel, O., Liu, M.Y.: Deep gaussian conditional random field network: A model-based deep network for discriminative denoising. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 4801–4809 (2016)
67. Wen, B., Li, Y., Pfister, L., Bresler, Y.: Joint adaptive sparsity and low-rankness on the fly: an online tensor reconstruction scheme for video denoising. In: IEEE International Conference on Computer vision (2017)
68. Xue, T., Chen, B., Wu, J., Wei, D., Freeman, W.T.: Video enhancement with task-oriented flow. International Journal of Computer Vision **127**(8), 1106–1125 (2019). DOI 10.1007/s11263-018-01144-2
69. Yang, D., Sun, J.: BM3D-Net: A convolutional neural network for transform-domain collaborative filtering. IEEE Signal Processing Letters **25**(1), 55–59 (2018). DOI 10.1109/LSP.2017.2768660
70. Yong, H., Huang, J., Hua, X., Zhang, L.: Gradient centralization: A new optimization technique for deep neural networks. CoRR **abs/2004.01461** (2020)
71. Zhang, K., Zuo, W., Chen, Y., Meng, D., Zhang, L.: Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. IEEE Transactions on Image Processing **26**(7), 3142–3155 (2017)
72. Zhang, K., Zuo, W., Zhang, L.: FFDNet: Toward a Fast and Flexible Solution for CNN based Image Denoising. CoRR **abs/1710.0** (2017)
73. Zhang, M., Lucas, J., Ba, J., Hinton, G.E.: Lookahead optimizer: k steps forward, 1 step back. In: Advances in Neural Information Processing Systems, pp. 9597–9608 (2019)
74. Zoran, D., Weiss, Y.: From learning models of natural image patches to whole image restoration. In: 2011 International Conference on Computer Vision, pp. 479–486 (2011). DOI 10.1109/ICCV.2011.6126278