12-2017

Recursive Non-Local Means Filter for Video Denoising

Redha A. Ali
University of Dayton, almahdir1@udayton.edu

Russell C. Hardie
University of Dayton, rhardie1@udayton.edu

Ali and Hardie *EURASIP Journal on Image and Video Processing* (2017) 2017:29
DOI 10.1186/s13640-017-0177-2

EURASIP Journal on Image and Video Processing

RESEARCH Open Access

Recursive non-local means filter for video denoising

Redha A. Ali and Russell C. Hardie*

Abstract

In this paper, we propose a computationally efficient algorithm for video denoising that exploits temporal and spatial redundancy. The proposed method is based on non-local means (NLM). NLM methods have been applied successfully in various image denoising applications. In the single-frame NLM method, each output pixel is formed as a weighted sum of the center pixels of neighboring patches, within a given search window. The weights are based on the patch intensity vector distances. The process requires computing vector distances for all of the patches in the search window. Direct extension of this method from 2D to 3D, for video processing, can be computationally demanding. Note that the size of a 3D search window is the size of the 2D search window multiplied by the number of frames being used to form the output. Exploiting a large number of frames in this manner can be prohibitive for real-time video processing. Here, we propose a novel recursive NLM (RNLM) algorithm for video processing. Our RNLM method takes advantage of recursion for computational savings, compared with the direct 3D NLM. However, like the 3D NLM, our method is still able to exploit both spatial and temporal redundancy for improved performance, compared with 2D NLM. In our approach, the first frame is processed with single-frame NLM. Subsequent frames are estimated using a weighted sum of pixels from the current frame and a pixel from the previous frame estimate. Only the single best matching patch from the previous estimate is incorporated into the current estimate. Several experimental results are presented here to demonstrate the efficacy of our proposed method in terms of quantitative and subjective image quality.

Keywords: Denoising, Video restoration, Non-local means, Recursive

1 Introduction

Digital videos are invariably corrupted by noise during acquisition. Digital video tends to have a lower signal-to-noise ratio (SNR) than static images, due to the short integration times needed to achieve desired frame rates [1]. Low-light conditions and small camera apertures tend to worsen the problem. In the case of some medical imagery, like x-ray images and video, short integration times are essential to limit the x-ray dose to the patient. While digital video may suffer from lower SNR, it also provides 3D data that often has significant temporal redundancy [2]. Video denoising algorithms seek to reduce noise by exploiting the both spatial and temporal correlation in the signal [1]. The non-local means (NLM) algorithm [3] for image denoising has received significant attention in the image processing community. This may be, in large part, because of generally good performance, and its intuitive and conceptually simple nature. The standard NLM algorithm is introduced by Buades et al. in [3]. The NLM method exploits self-similarity that appears in most natural images for noise reduction. In the single-frame NLM method, each output pixel is formed as a weighted sum of the center pixels of neighboring patches, within a given search window. The weights are based on similarity with respect to the reference patch. The similarity is measured by means of patch intensity vector distances. Pixels from patches with higher similarity (lower vector distances) are given more weight, using a negative exponential weighting. One or more tuning parameters are used to control the weighting.

Many variations of the NLM method have been proposed to reduce the computational complexity and/or improve the denoising performance. In the work of Mahmoudi and Sapiro [4], dissimilar neighborhoods are excluded from the weighted sum. Dissimilar blocks are identified based on mean value and gradient. This may improve performance, and it reduces the computational cost. Wang et al. [5] proposed an efficient summed square

*Correspondence: rhardie@udayton.edu
Department of Electrical and Computer Engineering, University of Dayton, 300 College Park, 45469-0226 Dayton, OH, USA

image scheme as another means to accelerate the patch similarity computations. A cluster tree arrangement has been used to group similar patches in [6]. A method using preselection of the most similar voxels, multithreading, and blockwise implementation is presented in [7]. Furthermore, adaptive smoothing neighborhoods are presented in [8], and a kernel regression method is presented in [9]. The method in [10] uses a spatially recursive moving-average filter to compute the Euclidean distances. The estimation of the mean square error (MSE) from a noisy image is performed using an analytical expression based on Stein's unbiased risk [11]. Another speed enhancement, based on probabilistic early termination, is proposed in [12]. Karnati et al. [13] proposed a multiresolution approach requiring fewer comparisons.

Many methods, originally proposed for single-image denoising, have been adapted to video denoising. Among these are the NLM method, which has been applied successfully to image sequences in [14] and [15]. Han et al. [16] introduced the dynamic non-local means (DNLM) video denoising method, which is based on Kalman filter theory. The basic idea of this filter is to use information from the past video frames to restore the current frame, combining the NLM and Kalman filtering algorithms. However, the computational complexity is still relatively high with this method. Another example of a 2D denoising method, later extended to 3D, is block-matching and 3D (BM3D) filtering [17]. BM3D generally outperforms NLM in single-image denoising but has a higher computational complexity because of the 3D transforms required. The BM3D method, like NLM, uses vector distances between 2D image blocks. The most similar blocks are stacked into a 3D group and then filtered through a transform-domain shrinkage operation. The BM3D filtering has been extended to video denoising in the video BM3D (VBM3D) algorithm described in [18]. While there may be many variations of patch-based image denoising algorithms. What they share in common is that they require computing vector distances between each reference patch and neighboring patches. Direct extension of such methods from 2D to 3D, for video processing, can be computationally demanding. Note that the size of a 3D search window is the size of the 2D search window multiplied by the number of frames being used to form the output. Exploiting a large number of frames in this manner can be prohibitive for real-time video processing.

In this paper, we propose a novel temporally recursive NLM (RNLM) algorithm for video processing. Our RNLM method takes advantage of temporal recursion for computational savings, compared with the direct 3D NLM. However, like the 3D NLM, our method is still able to exploit both spatial and temporal redundancy for improved performance, compared with 2D NLM. In our approach, the first frame is processed with single-frame NLM. Subsequent frames are estimated using a weighted sum of pixels from the current frame and a pixel from the previous frame estimate. Only the best-matching patch from the previous estimate is incorporated into the current estimate. This is done to maximize the temporal correlation. Our approach shares its recursive nature with DNLM in [16]. However, here, we have opted for a much simpler framework, in keeping with the simplicity of the original NLM. Several experimental results are presented here to demonstrate the efficacy of our proposed method in terms of quantitative and subjective image quality. We show that our approach offers a computationally simple approach to video denoising with a performance that rivals much more complex methods.

The remainder of the paper is organized as follows. Section 2 introduces the observation model and some benchmark methods. The proposed RNLM algorithm is presented in Section 3. Experimental results are presented in Section 4. Finally, we offer conclusions in Section 5.

2 Video restoration

2.1 Observation Model

In this section, we present the observation model and notation for our video restoration methods. Some of the key variables used in this paper are defined in Table 1. We use the standard degradation model, treating the noise as additive and signal independent. This is expressed as

$$y_k(i) = x_k(i) + n_k(i), \tag{1}$$

for $i = 1, 2, \ldots, N$ and $k = 1, 2, \ldots, K$. Note that $y_k(i)$ represents a pixel in the observed frame in the video sequence. The index i refers to the specific pixel in the spatial domain, and the k denotes the temporal frame number in the image sequence. The variable $x_k(i)$ denotes the corresponding pixel in the ideal input frame. The noise is represented by $n_k(i) \sim \mathcal{N}(0, \sigma_n^2)$, which is assumed to be samples of a zero-mean independent and identically distributed Gaussian random variable, with variance σ_n^2.

Table 1 Variable definitions used to describe the proposed video restoration method

$x_k(i)$	Ideal pixel i in frame k
$y_k(i)$	Noisy pixel i in frame k
$\hat{x}_k(i)$	Estimated pixel i in frame k
$\mathbf{y}_k(i)$	Lexicographical patch about pixel i in frame k in $\{y_k(\cdot)\}$
N	Number of pixels in one frame
K	Number of frames in input sequence
L	Number of frames used by 3D NLM to generate one frame output
M_s	NLM search window dimension ($M_s \times M_s$)
M_p	NLM patch dimension ($M_p \times M_p$)
N_s	BMA search window dimension ($N_s \times N_s$)
N_b	BMA block dimension ($N_b \times N_b$)

2.2 Single-frame non-local means filter

We begin by defining the single-frame NLM (SNLM) [3], upon which our method is built. Processing the frames from an image sequence individually, the SNLM output can be expressed as

$$\hat{x}_k(i) = \frac{1}{W_{k,i}} \sum_{j \in \varepsilon(i)} w_k(i,j) y_k(j), \tag{2}$$

where $\hat{x}_k(i)$ denotes the estimated image at pixel i in frame k. The set $\varepsilon(i)$ contains the indices of the pixels within an $M_s \times M_s$ search window centered about pixel i. The variable $w_k(i,j)$ is the weight applied to pixel j, when estimating pixel i in frame k. To normalize the weights, the variable $W_{k,i}$ is used, and this is simply the sum of the individual weights.

The SNLM weights are computed based on patch similarity and spatial proximity. In particular, $w_k(i,j)$ is computed as

$$w_k(i,j) = \exp\left\{ -\frac{\left\| \mathbf{y}_k(i) - \mathbf{y}_k(j) \right\|^2}{2\sigma_y^2} - \frac{d(i,j)^2}{2\sigma_d^2} \right\}, \tag{3}$$

and $W_{k,i}$ can be expressed as

$$W_{k,i} = \sum_{j \in \varepsilon(i)} w_k(i,j). \tag{4}$$

Note that the variable $\mathbf{y}_k(i)$ is a vector in lexicographical form containing pixels from an $M_p \times M_p$ patch centered about pixel i in frame k from the sequence $\{y_k(\cdot)\}$. The variable $d(i,j)^2$ is the squared Euclidean distance between pixel i and j. The parameter σ_y^2 is a tuning parameter to control the decay of the exponential weight function with regard to patch similarity, and σ_d^2 is a tuning parameter controlling the decay with regard to spatial proximity between pixels i and j. It can be seen from Eq. (3) that the weight given to pixel $y_k(j)$ goes down as $\left\| \mathbf{y}_k(i) - \mathbf{y}_k(j) \right\|^2$ goes up. The weight also goes down with the spatial distance between pixel i and j. Note that the tuning parameter, σ_y^2, is often set close to the noise variance. Studies of filter parameter selection can be found in [3, 19–21].

2.3 3D non-local means filter

In this section, we introduce a direct extension of 2D SNLM to 3D, to use as an additional performance benchmark. The 3D NLM uses a spatio-temporal search window to provide improved video denoising. In our approach, the patches remain 2D, but the search window is extended to 3D. This version of the 3D NLM is given by

$$\hat{x}_k(i) = \frac{1}{W_{k,i}} \sum_{j \in \varepsilon(i)} \sum_{m \in \psi(k)} w_{k,m}(i,j) y_m(j), \tag{5}$$

where $w_{k,m}(i,j)$ is the weight for pixel j of frame m, when estimating pixel i of frame k. The temporal search window is defined by $\psi(k)$, which is the set of frame indices used in the 3D search window for the estimation of frame k. In our experimental results, we use a causal temporal window comprised of the most recent L frames, and this is represented as $\psi(k) = \{k, k-1, \ldots, k-L+1\}$.

The 3D NLM weights are computed as

$$w_{k,m}(i,j) = \exp\left\{ -\frac{\left\| \mathbf{y}_k(i) - \mathbf{y}_m(j) \right\|^2}{2\sigma_y^2} - \frac{d(i,j)^2}{2\sigma_d^2} - \frac{(k-m)^2}{2\sigma_t^2} \right\}, \tag{6}$$

where a new temporal proximity tuning parameter, σ_t^2 is introduced. This extra tuning parameter is a natural extension to the spatial proximity parameter σ_d^2.
Samples from frame numbers that are far from the estimation sample will receive less weight using an exponential weighting controlled by σ_t^2. A small σ_t^2 gives a large penalty for frame difference. The weights are normalized using

$$W_{k,i} = \sum_{j \in \varepsilon(i)} \sum_{m \in \psi(k)} w_{k,m}(i,j). \tag{7}$$

Other similar extensions of the SNLM to 3D can be found in [22, 23].

3 Method

3.1 The proposed RNLM video denoising algorithm definition

The goal of the proposed RNLM method is to effectively exploit spatio-temporal information, as is done with the 3D NLM in Eq. (5) but with a computational complexity more in line with the SNLM in Eq. (2). To do so, RNLM estimate is formed as a weighted sum of pixels from the current frame, like Eq. (2), but it also includes a previous frame pixel estimate. That is, the current input frame and the prior output frame are used to form the current output. This type of temporal recursive processing helps to exploit the temporal signal correlation, without significantly increasing the search window size or overall computational complexity.

Specifically, the estimate for the proposed RNLM is given by

$$\hat{x}_k(i) = \frac{1}{W_{k,i}} \left[w_{\hat{x},k}(i) \hat{x}_{k-1}(s_k(i)) + \sum_{j \in \varepsilon(i)} w_{y,k}(i,j) y_k(j) \right], \tag{8}$$

where $\hat{x}_{k-1}(s_k(i))$ is the previous frame estimate (i.e., frame $k-1$) at pixel $s_k(i) \in \{1, 2, \ldots, N\}$. Pixel $s_k(i)$ is selected from $\{\hat{x}_{k-1}(\cdot)\}$ based on a standard block-matching algorithm (BMA) with respect to the block in input frame k centered about pixel i. For the selection of $s_k(i)$, we allow for a potentially different block and search size from that used for the within-frame processing. In

particular, the block-matching block size is $N_b \times N_b$, with an $N_s \times N_s$ search window. As in Eq. (2) for the SNLM, the set $\varepsilon(i)$ in Eq. (8) contains the indices of the pixels within an $M_s \times M_s$ search window centered about pixel i. The recursive weight in Eq. (8) is $w_{\hat{x},k}(i)$, and the non-recursive weights, $w_{y,k}(i,j)$, are similar to that for the SNLM. We shall define and discuss all of the weights shortly. The relationship among the various pixels used in the RNLM estimation process is depicted in Fig. 1. Shown are the raw unprocessed frames in parallel with the processed frames. Output $\hat{x}_k(i)$ is formed using a weighted sum of the input frames pixels, shown on the left, and the best matching previous processed output, shown on the back right.

Let us now define the weights. The non-recursive weights are defined in a manner similar to SNLM. Specifically, these are given by

$$w_{y,k}(i,j) = \exp\left\{-\frac{\|\mathbf{y}_k(i) - \mathbf{y}_k(j)\|^2}{h_{y_b}} - \frac{\sigma_n^2}{h_{y_n}}\right\}, \tag{9}$$

where h_{y_b} and h_{y_n} are tuning parameters. Here, we do not use the spatial distance weighting term of the SNLM. This could easily be added, but it did not provide improved performance in out experimental results. The recursive weights are given by

$$w_{\hat{x},k}(i) = \exp\left\{-\frac{\|\mathbf{y}_k(i) - \hat{\mathbf{x}}_{k-1}(s_k(i))\|^2}{h_{\hat{x}_b}} - \frac{\sigma^2_{\hat{x}_{k-1}(s_k(i))}}{h_{\hat{x}_n}}\right\}, \tag{10}$$

where $h_{\hat{x}_b}$ and $h_{\hat{x}_n}$ are tuning parameters, and $\sigma^2_{\hat{x}_{k-1}(s_k(i))}$ is the residual noise variance associated with $\hat{x}_{k-1}(s_k(i))$. The vector $\hat{\mathbf{x}}_{k-1}(s_k(i))$ is the $M_p \times M_p$ patch of pixels about pixel $\hat{x}_{k-1}(s_k(i))$ (shown in Fig. 1) in lexicographical vector form. The weight normalization factor here is given by

$$W_{k,i} = w_{\hat{x},k}(i) + \sum_{j \in \varepsilon_y(i)} w_{y,k}(i,j). \tag{11}$$

Finally, assuming the noise is independent and identically distributed, the residual noise variance can be computed recursively as follows

$$\sigma^2_{\hat{x}_k}(i) = \frac{w^2_{\hat{x},k}(i)\sigma^2_{\hat{x}_{k-1}}(s_k(i)) + \sum_{j \in \varepsilon_y(i)} w^2_{y,k}(i,j)\sigma_n^2}{W^2_{k,i}}. \tag{12}$$

Note that Eq. (12) is not the error variance. Rather it only accounts for the variance of the noise component of the error associated with the estimate $\hat{x}_{k-1}(s_k(i))$.

The RNLM weights in Eqs. (9) and (10) have a total of four tuning parameters, two to govern the non-recursive weights, and two to govern the recursive weights. In the non-recursive weights in Eq. (9), the parameter h_{y_b} serves the same role as σ_y^2 in the SNLM in Eq. (3). We refer to this weight as the non-recursive bias error weight. We view $||\mathbf{y}_k(i) - \mathbf{y}_k(j)||^2$ as a measure of the bias error in $y_k(j)$ with respect to the true sample $x_k(i)$ that we are estimating. That is, underlying signal differences between the

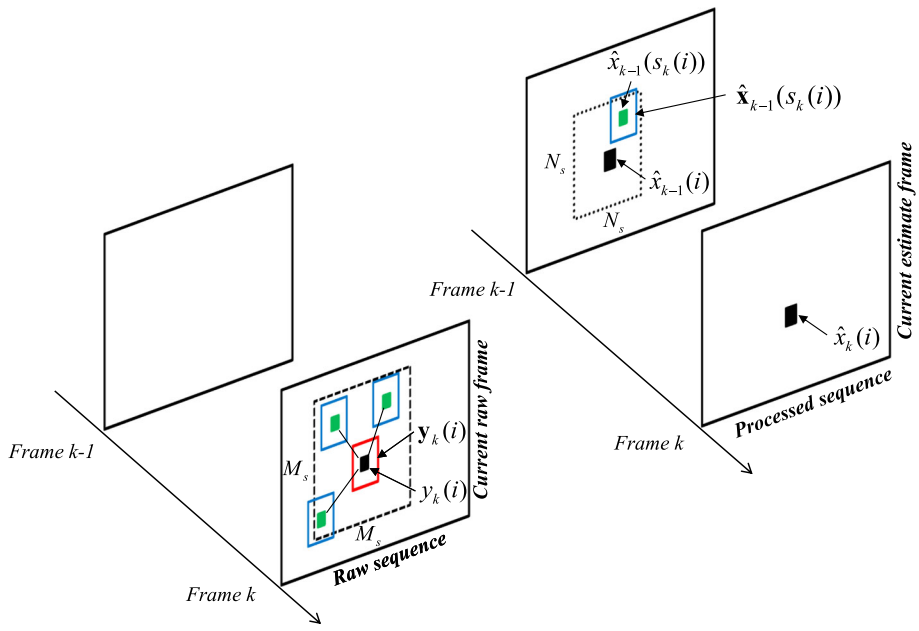

Fig. 1 Block diagram showing the pixels in the video sequence used in proposed RNLM algorithm. The *left side* shows the raw input frames, and the *right side* shows the processed frames. Because of the recursive processing, samples from the both input and processed frames are used to form the current estimate

pixels i and j are being quantified by this term. The tuning parameter h_{y_b} controls the exponential decay of the weights as a function of this bias error. The noise error associated with $y_k(j)$ is given by the constant noise variance σ_n^2. The noise variance in Eq. (9) is scaled by the h_{y_n}, to control the weight decay as a function of the noise variance. For the recursive weight, we have similar tuning parameters. The term $||\mathbf{y}_k(i)-\hat{\mathbf{x}}_{k-1}(s_k(i))||^2$ quantifies the bias error associated with $\hat{x}_{k-1}(s_k(i))$, and the residual noise variance associated with this estimate is $\sigma_{\hat{x}^2_{k-1}(s_k(i))}$, and may be computed using Eq. (12). We give each of these error quantities a tuning parameter, to balance their impact on the resulting filter weights. The bias error for the recursive sample is scaled by $h_{\hat{x}_b}$, and the residual noise term is scaled by $h_{\hat{x}_n}$.

We acknowledge that the proposed RNLM does not have the optimal framework of the Kalman filter [16]. However, it can exploit temporal signal correlation effectively, as we shall show in Section 4. By choosing the tuning parameters to balance the bias and noise components of the recursive and non-recursive terms, very good performance can be achieved with a low computational complexity. We believe that the RNLM may provide useful solution for many video denoising applications, and we believe it is in keeping with the spirit and simplicity of the original NLM method.

3.2 Computational complexity

In this section, we compare the computational complexity of SNLM, 3D NLM, and RNLM by counting the number of multiplications and additions required to compute one processed output pixel. Beginning with the SNLM, the number of floating point multiplies and adds to compute Eq. (2) is M_s^2 . Computing the weights based on Eq. (3) requires M_s^2 vector distances, for vectors of size $M_p^2 \times 1$. This requires approximately $2M_s^2M_p^2$ additions/subtractions, and $M_s^2M_p^2$ multiplications. Another M_s^2 adds and multiplies is needed to compute and apply the weight normalization in Eq. (4).

The 3D NLM algorithm requires LM_s^2 floating point multiplies and adds to compute Eq. (5). The weights from Eq. (6) require LM_s^2 vector distances using $M_p^2 \times 1$ vectors. This requires approximately $2LM_s^2M_p^2$ additions/subtractions and $LM_s^2M_p^2$ multiplications. Another LM_s^2 adds and multiplies is needed to compute and apply the weight normalization in Eq. (7). Thus, the complexity of the 3D NLM algorithm increases linearly with the number of frames, L.

The process of the RNLM filter is accomplished in several steps. The output in Eq. (8) requires $M_s^2 + 1$ floating point multiplies and adds. The computation of the weights in Eqs. (9) and (10) requires $M_s^2 + 1$ vector distances using $M_p^2 \times 1$ vectors. This requires approximately $2(M_s^2 + 1)M_p^2$ additions/subtractions and $2M_s^2M_p^2$ multiplications. The residual noise recursion in Eq. (12) requires $M_s^2 + 1$ floating point multiplies and adds. Finally, if BMA is used to find $s_k(i)$ in Eq. (8), this requires N_s^2 vector distances with $N_b^2 \times 1$ vectors. This requires approximately $2N_s^2N_b^2$ additions/subtractions and $N_s^2N_b^2$ multiplications. The weight normalization operation is accomplished with $M_s^2 + 1$ floating point adds to compute Eq. (11). Note that fast parallel architectures are available for rapid BMA computation [24–26].

Note that if we let $N_s = M_s$ and $N_b = M_p$, then RNLM has a computational complexity comparable to 3D NLM for $L = 2$ frames. Also, if we simply let $s_k(i) = i$ (i.e., no BMA option), the RNLM has a computational complexity that is approximately the same a SNLM. However, we have found that the BMA matching significantly improves performance in video sequences with significant amounts of motion. Furthermore, we have found that it is generally advantageous to choose $N_b > M_p$. This helps to provide a better match for the the important sample $\hat{x}_{k-1}(s_k(i))$. The size of the search window N_s maybe selected based on the temporal motion expected in the video sequence.

The RNLM method has a relatively low computational complexity compared with many transform-domain patch-based algorithms such as BM3D and VBM3D. Details of the BM3D implementation are presented in [27]. These transform-domain methods form a 3D cube from selected patches and perform a transform-domain shrinkage operation based on the fast Fourier transform (FFT), discrete cosine transform (DCT), or Wavelet transform. The patch selection operations are common to all of these methods, including the RNLM. However, the transform-based denoising methods have the additional requirement of computing forward and backward 3D transform on blocks of size $M = M_p \times M_p \times P$ for each pixel, where P is the number of patches selected [27]. Using row-column decomposition, such transforms are known to have a complexity of order $M\log(M)$ [27]. There is no such corresponding processing requirement for the RNLM.

4 Results and discussion

In order to illustrate the efficacy of the proposed RNLM algorithm, we present a number of experimental results. We compare our method to several state-of-the-art methods including SNLM [3], 3D NLM, BM3D [17], VBM3D [18], and DNLM [16]. Our results make use of standard and publicly available video sequences allowing for reproducible and comparative results. These sequences present variations in scene content, lighting conditions, and scene motion. We artifically degrade the imagery with Gaussian noise and compare the restored images to the originals.

Table 2 PSNR comparison with competitive denoising algorithms using five different sequences and eight noise standard deviation levels

	Video:	Salesman	Tennis	Fl. Gard.	Miss Am.	Foreman	
σ_n	Res.:	288 × 352	240 × 352	240 × 352	288 × 360	288 × 352	Overall PSNR
	Frames:	50	150	150	150	300	
5	SNLM	37.14	36.49	36.50	42.08	38.43	38.13
	BM3D	38.32	37.54	37.02	43.09	38.83	38.96
	RNLM (no BMA)	39.36	37.80	36.46	42.21	39.95	39.16
	RNLM (BMA)	39.59	38.61	37.09	42.74	40.55	39.71
	VBM3D	41.35	40.03	37.40	43.98	39.91	40.53
10	SNLM	33.28	32.66	31.03	39.23	35.17	34.27
	BM3D	34.51	33.63	31.85	40.39	35.21	35.12
	RNLM (no BMA)	36.14	34.04	31.05	39.66	36.13	35.41
	RNLM (BMA)	36.31	34.40	32.08	40.22	36.72	35.95
	VBM3D	38.33	36.37	32.58	42.17	36.67	37.22
15	SNLM	30.92	30.57	28.16	37.22	33.16	32.01
	BM3D	32.43	31.65	28.99	38.61	33.26	32.99
	RNLM (no BMA)	33.16	31.96	28.50	37.66	34.20	33.08
	RNLM (BMA)	34.30	32.49	29.47	38.53	34.85	33.93
	VBM3D	36.55	34.35	29.93	40.93	34.87	35.33
20	SNLM	29.38	29.35	26.29	35.69	31.70	30.48
	BM3D	31.02	30.35	27.04	37.22	31.90	31.51
	RNLM (no BMA)	31.44	30.65	26.62	36.61	32.68	31.59
	RNLM (BMA)	32.79	30.93	27.74	37.24	33.37	32.42
	VBM3D	35.07	32.92	28.06	39.93	33.54	33.91
25	SNLM	28.20	28.52	24.91	34.51	30.48	29.32
	BM3D	29.94	29.38	25.57	36.06	30.85	30.36
	RNLM (no BMA)	30.16	29.26	25.30	35.57	31.08	30.27
	RNLM (BMA)	31.61	29.92	26.57	36.40	32.09	31.32
	VBM3D	33.69	31.83	26.55	39.04	32.42	32.71
30	SNLM	27.27	27.69	23.83	33.54	29.41	28.35
	BM3D	29.06	28.58	24.39	34.19	29.99	29.24
	RNLM (no BMA)	29.22	28.28	24.23	34.79	30.08	29.32
	RNLM (BMA)	30.64	28.80	25.55	35.70	31.10	30.36
	VBM3D	32.45	30.96	25.21	37.40	31.42	31.49
35	SNLM	26.53	26.85	22.96	32.72	28.47	27.51
	BM3D	28.28	27.85	23.41	34.19	29.23	28.59
	RNLM (no BMA)	28.48	27.53	23.35	34.05	29.53	28.58
	RNLM (BMA)	29.82	27.99	24.68	35.11	30.27	29.57
	VBM3D	31.34	30.24	24.28	37.40	30.59	30.77
40	SNLM	25.93	26.07	22.23	32.02	27.65	26.78
	BM3D	27.38	27.04	22.60	33.26	28.30	27.72
	RNLM (no BMA)	27.96	25.74	22.55	33.55	28.83	27.72
	RNLM (BMA)	29.16	27.16	23.96	34.58	29.54	28.87
	VBM3D	30.32	29.60	23.28	36.33	29.74	29.85

Table 3 SSIM comparison with published results for several competitive denoising algorithms using five different sequences and three noise standard deviation levels

σ_n	Video:	Salesman	Tennis	Fl. Gard.	Miss Am.	Foreman	Overall SSIM
	Res.:	288 × 352	240 × 352	240 × 352	288 × 360	288 × 352	
	Frames:	50	150	150	150	300	
10	SNLM	0.887	0.853	0.934	0.890	0.907	0.894
	BM3D	0.917	0.869	0.963	0.959	0.918	0.925
	WRSTF [31]	0.932	0.897	0.953	0.908	0.927	0.923
	SEQWT [32]	0.900	0.842	0.941	NA	NA	0.894
	3DWTF [33]	0.923	0.856	0.909	NA	NA	0.896
	IFSM [34]	0.904	0.855	0.927	0.904	0.886	0.895
	3DSWDCT [36]	0.955	0.894	0.959	0.946	0.932	0.937
	VBM3D	0.959	0.916	0.963	0.967	0.934	0.948
	ST-GSM [36]	0.960	0.894	0.950	0.952	0.937	0.939
	DNLM [16]	0.931	0.856	0.947	0.964	0.946	0.928
	RNLM (no BMA)	0.937	0.881	0.953	0.954	0.923	0.930
	RNLM (BMA)	0.939	0.895	0.960	0.960	0.925	0.936
15	SNLM	0.825	0.759	0.886	0.829	0.870	0.834
	BM3D	0.878	0.817	0.937	0.948	0.888	0.893
	WRSTF [31]	0.901	0.839	0.922	0.877	0.877	0.883
	SEQWT [32]	0.846	0.722	0.893	NA	NA	0.820
	3DWTF [33]	0.903	0.793	0.872	NA	NA	0.856
	IFSM [34]	0.851	0.776	0.882	0.857	0.836	0.840
	3DSWDCT [36]	0.930	0.834	0.931	0.928	0.907	0.906
	VBM3D	0.943	0.874	0.940	0.961	0.911	0.926
	ST-GSM [36]	0.941	0.841	0.925	0.943	0.917	0.913
	DNLM [16]	0.889	0.795	0.906	0.951	0.929	0.894
	RNLM (no BMA)	0.883	0.811	0.900	0.941	0.887	0.884
	RNLM (BMA)	0.902	0.829	0.933	0.945	0.901	0.902
20	SNLM	0.768	0.679	0.836	0.761	0.833	0.775
	BM3D	0.843	0.780	0.909	0.936	0.861	0.866
	WRSTF [31]	0.868	0.790	0.889	0.846	0.873	0.853
	SEQWT [32]	0.796	0.716	0.842	NA	NA	0.785
	3DWTF [33]	0.882	0.740	0.840	NA	NA	0.821
	IFSM [34]	0.801	0.709	0.837	0.812	0.793	0.790
	3DSWDCT [36]	0.905	0.790	0.900	0.909	0.884	0.878
	VBM3D	0.923	0.836	0.918	0.956	0.891	0.905
	ST-GSM [36]	0.923	0.797	0.900	0.936	0.901	0.891
	DNLM [16]	0.849	0.758	0.865	0.939	0.913	0.865
	RNLM (no BMA)	0.881	0.779	0.881	0.930	0.855	0.865
	RNLM (BMA)	0.886	0.783	0.905	0.937	0.876	0.877

NA not applicable

Table 4 Additional PSNR performance comparisons with various published results. All PSNR values are reported in decibel

Video	Salesman		Fl. Gard.	Miss Am.	Suzie		Foreman	
Input PSNR (dB)	28	24	28	28	28	24	28	24
STA [23]	35.13	32.60	31.33	39.39	37.07	35.11	34.94	32.90
K-SVD [37]	37.91	35.59	32.13	40.49	37.96	35.95	37.86	35.86
ST-GSM [36]	37.93	35.17	NA	41.43	38.36	36.21	36.85	34.37
3D-Patch [22]	39.26	36.35	NA	42.23	38.40	36.32	36.88	34.55
VBM3D [18]	38.79	36.07	32.51	41.64	38.16	36.24	37.27	35.19
SNLM [16]	32.97	30.02	30.33	38.47	34.33	31.90	34.92	32.65
DNLM [16]	35.22	32.73	31.28	39.70	37.22	35.25	36.19	34.06
RNLM (no BMA)	36.19	32.78	30.94	39.60	38.43	35.83	36.17	33.87
RNLM (BMA)	36.36	34.03	32.01	40.07	39.36	36.55	36.46	34.37

The metrics we shall use are the peak signal-to-noise ratio (PSNR) and the structural similarity index (SSIM). The PSNR metric in units of decibels (dB) is defined as

$$\mathrm{PSNR}(k) = 10 \times \log_{10}\left(\frac{R^2}{MSE(k)}\right), \tag{13}$$

where R is the maximum limit of the dynamic range of the image. For our 8-bit images, $R = 255$. The variable, $MSE(k)$, is the mean squared error for frame k, given by

$$MSE(k) = \frac{1}{N}\sum_{i=1}^{N}(x_k(i) - \hat{x}_k(i))^2, \tag{14}$$

where $x_k(i)$ is the true pixel value, and $\hat{x}_k(i)$ is the estimated pixel. The SSIM provides an aditional metric that

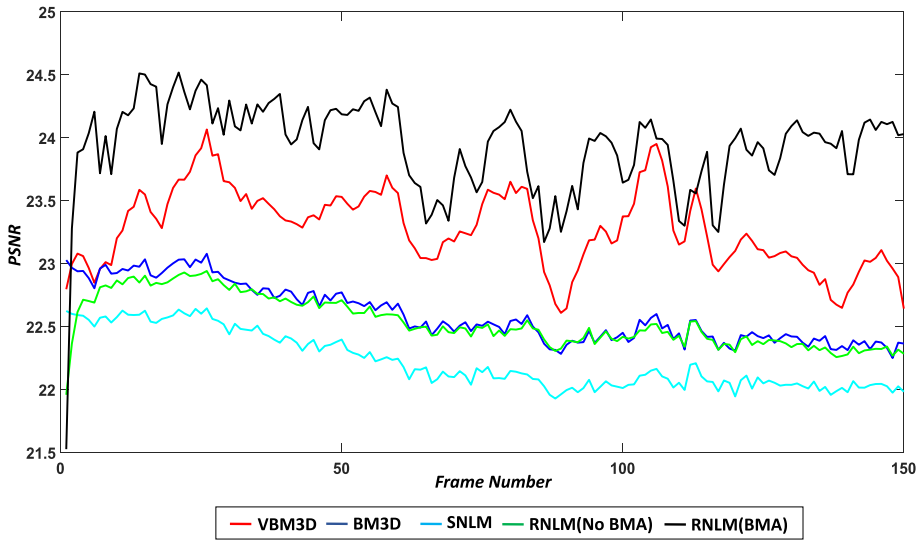

Fig. 2 PSNR (dB) versus frame number using several methods for the Flower Garden sequence corrupted with a noise level of $\sigma = 40$. The RNLM (BMA) method gives the best performance for this sequence

some argue is more consistent with subjective perception than PSNR [28–30].

In Table 2, we provide PSNR results for five different image sequences, each with eight different noise standard deviations. The proposed RNLM method results are reported for two variations. The results labeled RNLM (BMA) refers to the proposed method where block matching is employed to find the best match for the recursive sample $\hat{x}_{k-1}(s_k(i))$. The (no BMA) version simply uses the estimate pixel position, such that $s_k(i) = i$. This version has a reduced computational complexity. However, as shown in Table 2, using BMA gives improved results.

The results in Table 2 show that the RNLM method consistently outperforms the SNLM. Thus, the recursive processing is providing a clear performance benefit. RNLM also outperforms single-frame BM3D. VBM3D does provide higher PSNR values in most cases, but the computational complexity of that method is far greater, and the processing of the video is non-causal. On the other hand, RNLM has a low computational complexity, and the processing is fully causal, allowing for real-time processing.

SSIM results for the same sequences are shown in Table 3. In this table, several additional published results are reported for comparison. These include wavelet domain recursive spatio-temporal filtering (WRSTF) [31], sequential wavelet domain and temporal filtering (SEQWT) [32], 3D wavelet transform method (3DWTF) [33], inter-frame statistical modeling (IFSM) [34], 3D sliding window discrete cosine transform (3DSWDCT) [35], VBM3D [18], spatiotemporal Gaussian scale mixture (ST-GSM) [36], and DNLM [16]. The results for these methods are the reported results from the respective papers for the same sequences and noise levels. The standard deviations of the additive Gaussian noise are 10, 15, and 20. It can be seen that the RNLM method gives higher SSIM than SNLM, WRSTF, SEQWT, 3DWTF, IFSM, and DNLM, with average SSIM gains of 0.070, 0.018, 0.059, 0.034, 0.063, and 0.0091, respectively. The SSIM of RNLM is competitive with 3DSWDCT and ST-GSM. However, VBM3D gives the best results here.

Additional PSNR results are provided in Table 4 for two noise levels. The noisy input image PSNRs here are 24 and 28 dB. Comparison methods here include space time adaption (STA) [23], K-means singular value decomposition (K-SVD) [37], ST-GSM [36], 3D-patch [22], VBM3D [18], DNLM [16], and SNLM. The NLM patch size is $M_p = 7$. The NLM search window $M_s = 11$. The BMA search and block-matching size used $N_s = 3$, $N_b = 29$, respectively. As one can see, RNLM (BMA) outperforms STA, DNLM, RNLM (No BMA), and SNLM with the average PSNR gains of 1.35, 0.95, 0.68, and 2.96, respectively.

Figures 2 and 3 show the restored PSNR for individual frames 1 to 150 for the sequences Flower Garden and Salesman, respectively. The noise standard deviation for these results is $\sigma_n = 40$. The methods shown are VBM3D, BM3D, SNLM, RNLM (No BMA), and RNLM (BMA). The proposed RNLM (BMA) provides the best results in the Flower Garden video sequence, and provides the second best performance on the Salesman sequence.

In Fig. 4, we offer a visual comparison of denoised frame 92 from the Salesman sequence with Gaussian noise at $\sigma_n = 40$. Figure 4a shows the original frame. The noisy frame is shown in Fig. 4b. Figure 4c is the denoised image

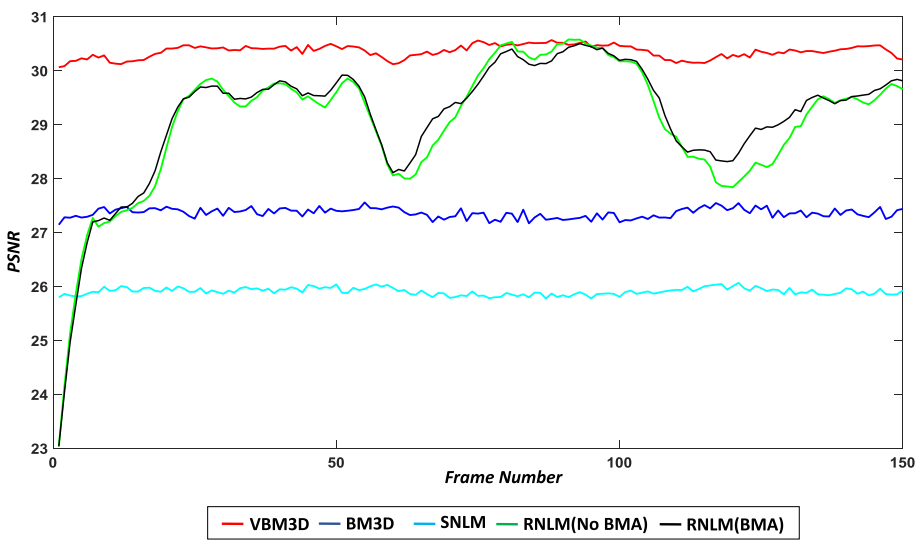

Fig. 3 PSNR (dB) versus frame number using several methods for the Salesman sequence corrupted with a noise level of $\sigma = 40$. With no background motion, RNLM with and without BMA perform about the same. VBM3D gives the best performance for this sequence

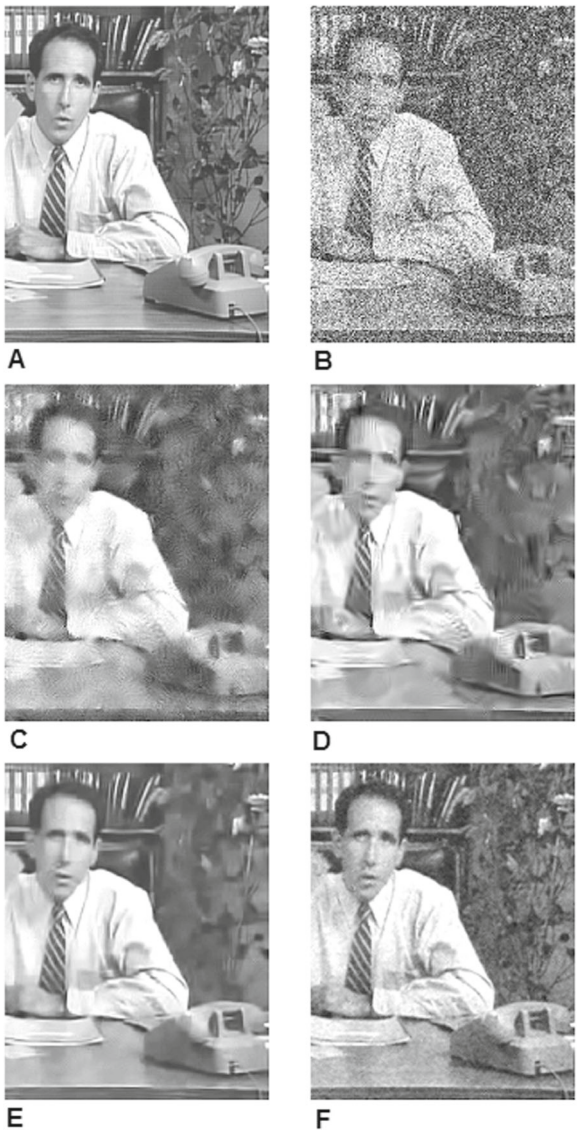

Fig. 4 Comparison of denoised frame 92 from the Salesman sequence with $\sigma_n = 40$. **a** Original frame. **b** Corrupted frame. **c** SNLM. **d** BM3D. **e** VBM3D. **f** RNLM (BMA). The RNLM method appears to preserve the background details better than the other methods

using SNLM. This method struggles to effectively handle the high levels of noise. However, with decreased σ_n, it tends to exhibit much better subjective visual performance. Figure 4d is the BM3D denoised frame. While it successfully reduces the noise, here, it produces an over-smooth result and detail such as the mouth, eyes, and texture of the original frame is lost. Figure 4e shows the VBM3D denoised frame. This result does a good job with noise reduction and some detail preservation. However, it also tends to over-smooth texture in this image, such as the plant leaves. Finally, Fig. 4f shows the output of the proposed RNLM (BMA) algorithm. We believe that it produces a visually pleasing result here, with a good balance of noise reduction and detail/texture preservation.

5 Conclusions

In this paper, we have presented a new temporally recursive NLM algorithm for video denoising. The output of the new RNLM method is a weighted sum of pixels from the current noisy frame, and of a selected pixel from the prior processed frame. This type of recursive processing allows us to exploit temporal correlation with little additional computational cost, compared with a SNLM. We have explored two versions of the RNLM method here, RNLM (BMA) that uses BMA for motion compensation and RNLM (No BMA) that simply uses the previous pixel at the same location to be included in the weighted sum. The results also show that RNLM (BMA) consistently outperformed the RNLM (No BMA). The computational complexity of the RNLM (BMA) method is approximately the same as 3D-NLM with just two frames. Both versions of RNLM method consistently outperform SNLM. Furthermore, our results show that RNLM (BMA) is competitive with much more computationally complex algorithms, such as BM3D and VBM3D. We believe the proposed method offers a simple and practical video denoising solution, capable of balancing noise reduction and detail preservation. Because of its low computational cost, we believe it is well suited for real-time implementation.

Acknowledgements
The authors would like to thank Dr. Khaled Mohamed for his assistance in the initial coding and testing of the RNLM method.

Funding
This work was the result of an unfunded academic research project. The authors have no funding sources to declare.

Availability of data and materials
The video sequences used in this paper may be found in the following database: http://see.xidian.edu.cn/vipsl/database_Video.html.

Authors' contributions
The two authors contributed equally to the presented research and writing of this paper. The authors have read and approved the final manuscript.

Authors' information
Redha Almahdi is a Ph.D. student in the department of Electrical and Computer Engineering at the University of Dayton. He received his MS degree in Electrical Engineering from the same department in 2016, and a Bachelor of Science in Computer Engineering from the College of Electronics Technology, Bani Walid, in 2012.
Russell C. Hardie is a Full Professor in Department of Electrical and Computer Engineering at the University of Dayton, and holds a joint appointment in the Department of Electro-Optics and Photonics. Dr. Hardie received the University of Dayton's top university-wide teaching award, the 2006 Alumni Award in Teaching, presented in the name of the Alumni Association. Along with several collaborators, Dr. Hardie received the Rudolf Kingslake Medal and Prize from SPIE in 1998 for work on image super-resolution algorithms. In 1999, he received the School of Engineering Award of Excellence in Teaching at the University of Dayton. Dr. Hardie's research interests include a wide range of topics in the area of image and video processing. His focus has been in the area of image restoration and enhancement.

Competing interests
The authors declare that they have no competing interests.

Consent for publication
Not applicable.

Ethics approval and consent to participate
Not applicable.

Publisher's Note

Received: 27 September 2016 Accepted: 23 March 2017
Published online: 17 April 2017

References

1. Z Lin, X Li, Z Sun, in *International Symposium on Computers & Informatics (ISCI)*. The summary of video denoising method (2015 International Symposium on Computers & Informatics, Amsterdam, 2015). http://www.atlantis-press.com/php/pub.php?publication=isci-15
2. G Chen, J Zhang, D Li, H Chen, Robust kronecker product video denoising based on fractional-order total variation model. Sig. Process. **119**, 1–20 (2016)
3. A Buades, B Coll, J-M Morel, A review of image denoising algorithms, with a new one. Multiscale Model. Simul. **4**(2), 490–530 (2005)
4. M Mahmoudi, G Sapiro, Fast image and video denoising via nonlocal means of similar neighborhoods. IEEE Signal Process. Lett. **12**(12), 839–842 (2005)
5. J Wang, Y Guo, Y Ying, Y Liu, Q Peng, in *Image Processing, 2006 IEEE International Conference On*. Fast non-local algorithm for image denoising (IEEE, 2006), pp. 1429–1432. http://ieeexplore.ieee.org/abstract/document/4106808/
6. T Brox, O Kleinschmidt, D Cremers, Efficient nonlocal means for denoising of textural patterns. IEEE Trans. Image Process. **17**(7), 1083–1092 (2008)
7. Coupé, P Yger, S Prima, P Hellier, C Kervrann, C Barillot, An optimized blockwise nonlocal means denoising filter for 3-d magnetic resonance images. IEEE Trans. Med. Imaging. **27**(4), 425–441 (2008)
8. C Kervrann, J Boulanger, Optimal spatial adaptation for patch-based image denoising. IEEE Trans. Image Process. **15**(10), 2866–2878 (2006)
9. P Chatterjee, P Milanfar, in *Electronic Imaging 2008*. A generalization of non-local means via kernel regression (International Society for Optics and Photonics. Proc. SPIE 6814, Computational Imaging VI, 68140P (February 26, 2008), 2008), p. 68140P. doi:10.1117/12.778615. http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=812032
10. B Goossens, Q Luong, A Pizurica, W Philips, in *2008 International Workshop on Local and Non-Local Approximation in Image Processing (LNLA 2008)*. An improved non-local denoising algorithm (2008 International Workshop on Local and Non-Local Approximation in Image Processing, 2008), pp. 143–156. https://biblio.ugent.be/publication/517393/file/1046804
11. D Van De Ville, M Kocher, Sure-based non-local means. IEEE Signal Process. Lett. **16**(11), 973–976 (2009)
12. R Vignesh, BT Oh, C-CJ Kuo, Fast non-local means (nlm) computation with probabilistic early termination. IEEE Signal Process. Lett. **17**(3), 277–280 (2010)
13. V Karnati, M Uliyar, S Dey, in *Image Processing (ICIP), 2009 16th IEEE International Conference On*. Fast non-local algorithm for image denoising (IEEE, 2009), pp. 3873–3876. http://ieeexplore.ieee.org/abstract/document/5414044/
14. A Buades, B Coll, JM Morel, in *Advanced Video and Signal Based Surveillance, 2005. AVSS 2005. IEEE Conference On*. Denoising image sequences does not require motion estimation (IEEE, 2005), pp. 70–74. http://ieeexplore.ieee.org/abstract/document/1577245/
15. A Buades, B Coll, J-M Morel, Nonlocal image and movie denoising. Int. J. Comput. Vis. **76**(2), 123–139 (2008)
16. Y Han, R Chen, Efficient video denoising based on dynamic nonlocal means. Image Vision Comput. **30**(2), 78–85 (2012)
17. K Dabov, A Foi, V Katkovnik, K Egiazarian, in *Electronic Imaging 2006*. Image denoising with block-matching and 3d filtering (International Society for Optics and Photonics. Proc. SPIE 6064, Image Processing: Algorithms and Systems, Neural Networks, and Machine Learning, 606414 (February 17, 2006), 2006), p. 606414. doi:10.1117/12.643267. http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=728740
18. K Dabov, A Foi, K Egiazarian, in *European Signal Processing Conference. (Vol. 149)*. Video denoising by sparse 3d transform-domain collaborative filtering (European Signal Processing Conference 2007, Tampere, 2007), pp. 145–149. http://www.cs.tut.fi/~foi/GCF-BM3D/VBM3D_EUSIPCO_2007.pdf
19. Z Wang, L Zhang, An adaptive fast non-local image denoising algorithm. J. Image Graphics. **14**(4), 669–675 (2009)
20. JV Manjón, J Carbonell-Caballero, JJ Lull, García-Martí, G, Martí-Bonmatí, L, M Robles, Mri denoising using non-local means. Med. Image Anal. **12**(4), 514–523 (2008)
21. H Li, CY Suen, A novel non-local means image denoising method based on grey theory. Pattern Recognit. **49**, 237–248 (2016)
22. X Li, Y Zheng, Patch-based video processing: a variational bayesian approach. IEEE Trans. Circ. Syst. Video Technol. **19**(1), 27–40 (2009)
23. J Boulanger, C Kervrann, P Bouthemy, Space-time adaptation for patch-based image sequence restoration. IEEE Trans. Pattern Anal. Mach. Intell. **29**(6), 1096–1102 (2007)
24. C Yan, Y Zhang, J Xu, F Dai, J Zhang, Q Dai, F Wu, Efficient parallel framework for hevc motion estimation on many-core processors. IEEE Trans. Circ. Syst. Video Technol. **24**(12), 2077–2089 (2014). doi:10.1109/TCSVT.2014.2335852
25. C Yan, Y Zhang, J Xu, F Dai, L Li, Q Dai, F Wu, A highly parallel framework for hevc coding unit partitioning tree decision on many-core processors. IEEE Signal Process. Lett. **21**(5), 573–576 (2014). doi:10.1109/LSP.2014.2310494
26. C Yan, Y Zhang, F Dai, J Zhang, L Li, Q Dai, Efficient parallel hevc intra-prediction on many-core processor. Electron. Lett. **50**(11), 805–806 (2014). doi:10.1049/el.2014.0611
27. M Lebrun, An analysis and implementation of the BM3D image denoising method. Image Process. Line. **2**, 175–213 (2012). doi:10.5201/ipol.2012.l-bm3d
28. Z Wang, AC Bovik, A universal image quality index. IEEE Signal Process. Lett. **9**(3), 81–84 (2002)
29. Z Wang, AC Bovik, HR Sheikh, EP Simoncelli, Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)
30. KM Mohamed, RC Hardie, A collaborative adaptive wiener filter for image restoration using a spatial-domain multi-patch correlation model. EURASIP J. Adv. Signal Process. **2015**(1), 1–23 (2015)
31. V Zlokolica, Pižurica, W Philips, Wavelet-domain video denoising based on reliability measures. IEEE Trans. Circ. Syst. Video Technol. **16**(8), 993–1007 (2006)
32. A Pizurica, V Zlokolica, W Philips, in *Advanced Video and Signal Based Surveillance, 2003. Proceedings. IEEE Conference on*. Combined wavelet domain and temporal video denoising (IEEE, 2003), pp. 334–341. http://ieeexplore.ieee.org/abstract/document/1217940/
33. IW Selesnick, KY Li, in *Optical Science and Technology, SPIE's 48th Annual Meeting*. Video denoising using 2d and 3d dual-tree complex wavelet transforms (International Society for Optics and Photonics. Proc. SPIE 5207, Wavelets: Applications in Signal and Image Processing X, 607 (November 14, 2003), 2003), pp. 607–618. doi:10.1117/12.504896. http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=827225
34. SM Rahman, MO Ahmad, M Swamy, Video denoising based on inter-frame statistical modeling of wavelet coefficients. IEEE Trans. Circ. Syst. Video Technol. **17**(2), 187–198 (2007)
35. D Rusanovskyy, K Egiazarian, in *International Conference on Advanced Concepts for Intelligent Vision Systems*. Video denoising algorithm in sliding 3D DCT domain (Springer, Berlin Heidelberg, 2005), pp. 618–625. https://link.springer.com/chapter/10.1007/11558484_78
36. G Varghese, Z Wang, Video denoising based on a spatiotemporal gaussian scale mixture model. IEEE Trans. Circ. Syst. Video Technol. **20**(7), 1032–1040 (2010)
37. M Protter, M Elad, Image sequence denoising via sparse and redundant representations. IEEE Trans. Image Process. **18**(1), 27–35 (2009)