EURASIP Journal on Advances in Signal Processing

RESEARCH Open Access

Optimal graph edge weights driven nlms with multi-layer residual compensation

Fang Yang*, Xin Chen and Li Chai

*Correspondence: yangfang.idif@wust.edu.cn
Engineering Research Center of Metallurgical Automation and Measurement Technology, Wuhan University of Science and Technology, Wuhan 430081, Hubei, China

Abstract

Non-local Means (NLMs) play essential roles in image denoising, restoration, inpainting, etc., due to its simple theory but effective performance. However, when the noise increases, the denoising accuracy of NLMs decreases significantly. This paper further develop the NLMs-based denoising method to remove noise with less loss of image details. It is realized by embedding an optimal graph edge weights driven NLMs kernel into a multi-layer residual compensation framework. Unlike the patch similarity-based weights in the traditional NLMs filters, the edge weights derived from the optimal graph Laplacian regularization consider (1) the distance between the target pixel and the candidate pixel, (2) the local gradient and (3) the patch similarity. After defining the weights, the graph-based NLMs kernel is then put into a multi-layer framework. The corresponding primal and residual terms at each layer are finally fused with learned weights to recover the image. Experimental results show that our method is effective and robust, especially for piecewise smooth images.

Keywords: Non-local means, Image denoising, Graph signal processing, Multi-layer, Residual compensation

1 Introduction

Image denoising is one of the most fundamental and important tasks in image processing and computer vision. Generally speaking, it aims at retrieving the clean image $u_{ori} \in \mathbb{R}^{S_1 \times S_2}$ from an observed noisy image $u \in \mathbb{R}^{S_1 \times S_2}$. Typically, the noisy image is assumed to be corrupted by Gaussian noise. In the past decades, numerous denoising methods have been proposed. Regarding the way to separate u_{ori} from u, the denoising methods can be divided into two classes: those implemented in the spatial domain and those implemented in the transform domain.

In the spatial domain, classical methods denoise an image by averaging the pixels with different weights, e.g., equal weights of box-car filter, weights depending on the distance between pixels in Gaussian filter, weights computed from geometric and radiometric distances of bilateral filter [11, 12, 19]. Besides, various extensions have been proposed to balance the smoothness and details, e.g., averaging in local windows with adaptive size [22] or local regions with adaptive shape [32]. In contrast to these connected local regions, Buades et al. proposed a method to average pixels in non-local regions named non-local means (NLMs) [3]. The main idea of NLMs is to select similar pixels

in a non-local region (even the whole image), then average them with different weights. Since the similarity between two pixels is computed by the corresponding non-local patches around them, NLMs are robust to noise and yield effective performance. However, the NLMs filter removes some image details such as edges and rare textures during denoising, especially when the SNR is low. Although total variation (TV) regularization models [7, 17, 26] have been combined with NLMs approaches to deal with rare patches (no similar pixels have been selected), it still can not address this issue.

The basic idea of denoising in the transform domain is to separate noise from the observed signal in the transform domain. In general, the noise is randomly distributed, changes rapidly in images, and has almost uniform power across the whole frequency domain. The clean images usually change slowly in local areas, and their power is generally distributed on low frequencies. Based on this phenomenon, various transforms have been used for denoising, e.g., Curvelet transform [25], Wavelet transform [6], graph Fourier transform [15, 18, 29]. Among all transform-based methods, the block-match in 3D transform-domain filter (BM3D) is the most popular and widely used method [8]. BM3D filter is effective by combining the NLMs theory with the wavelet transform-based denoising. Except for the transforms with fixed bases, data-driven transforms have also been widely used in image denoising tasks, including PCA [2], sparse coding [14], dictionary learning [9] and compressed sensing [10].

More recently, machine learning-based denoising methods, especially deep learning-based approaches, attract public attention. The deep network was first applied in image denoising in [16], in which the auto-encoder network does not need manually set parameters for removing the noise. Then Zhang et al. proposed the DnCNN to deal with image denoising, super-resolution, and JPEG image deblocking [34]. The generative adversarial network (GAN) is used to remove blind noise in [4]. In addition, attention mechanism [30] and batch re-normalization [31] theories have been introduced in denoising tasks, which achieve excellent performance. In a word, the recent deep learning approaches can yield better results than the traditional filters, however, a considerable amount of high-quality training data is required for the network training, which is not always available in reality.

This paper aims at developing the NLMs by introducing the graph signal processing theory and a multi-layer framework. Graph signal processing (GSP) is a powerful and developed tool for analyzing signals on graphs [5, 20, 24, 33]. Traditional image processing methods regard the image domain as regular 2D grids. But if one treats each pixel on an image as a node of a graph and constructs proper links between nodes, one can interpret an image as a signal on an irregular graph. Then the computation of similarity between pixels and patches will no longer be restricted by the regular grids. In this way, the image information can be exploited more comprehensively by using the GSP. The optimal graph Laplacian regularization (OGLR) method derives the optimal metric space in the sense of minimum mean square error (MMSE), thus defining the optimal edge weights. Unlike the patch similarity-based weights in the classic NLMs filters, the OGLR defines the weights by considering (1) the distance between the target pixel and the candidate pixel, (2) the local gradient and (3) the patch similarity. The OGLR method yields good results for image denoising. However, it needs a large number of iterations to achieve comparable performance. Moreover, it involves a lot of matrix inversion, making

the method time costly. Hence in our paper, we replace the weights in the classical NLMs by the graph edge weights from the OGLR algorithm and embed the newly obtained NLMs into a multi-layer framework. The main contributions of this paper are threefold:

Firstly, our method uses the edge weight defined on a graph structure to compute the similarity between nodes and patches, which behaves better than the traditional NLMs. Experiments show that our method is comparable with the state-of-the-art methods, both visually and quantitatively. Furthermore, our method is better at denoising piecewise smooth images, especially when the noise level is high.

Secondly, a multi-layer representation is performed in our method to remove the noise while preserving the details. Obviously, the multi-layer strategy helps to smooth the image better. In addition, for the sake of recovering image details, the residual terms at each layer (the difference between the input and output of the NLMs filter) are combined with the smooth filtered image.

Last but not least, the coefficients of each component derived from each layer, including the smooth filtered image and the residual terms, are learned according to the least square method. These coefficients can be set as default parameters for any image.

Note that a similar idea has been proposed in [28]. However, the key difference is that our method adapts the NLMs filter parameters (the graph Laplacian regularization) to the input image, instead of a fixed filter presented in [28].

This paper is organized as follows. Section 2 introduces the related work about graph construction, the OGLR algorithm and the multi-layer representation of filter images. The proposed denoising method is detailed in Sect. 3. Experiments and results are presented in Sects. 4 and 5 respectively. And finally, conclusions are given in Sect. 6.

2 Related work

2.1 Graph construction

Let $\mathcal{G}(\mathcal{V},\mathcal{E})$ denote a graph structure, $\mathcal{V}=\{v_i\}_{i=1}^{N}$ is a set of N nodes, and $\mathcal{E}=\{e_{ij}\}$ a set of edges. If two different nodes v_i and v_j are connected, there exists an edge weight $w_{i,j}$ describing the affinity between these two nodes. Generally, the larger the $w_{i,j}$ is, the more similar or correlated the nodes v_i and v_j are. A widely used form of the weight $w_{i,j}$ is as follows:

$$w_{i,j}=\begin{cases}\exp\left(-\frac{\phi_{ij}^2}{2\varepsilon^2}\right), & \text{if}\left|\phi_{ij}\right|\le r,\\ 0, & \text{otherwise,}\end{cases} \tag{1}$$

where ϕ_{ij} measures the distance between two nodes v_i and v_j, and r is a threshold. Note that ϕ_{ij} does not necessarily correspond to the Euclidean distance between the nodes. Typically, w_{ij} is non-negative. Apparently, the larger the distance between two nodes, the smaller $w_{i,j}$ is. The weighted affinity matrix $\mathbf{W}\in\mathbb{R}^{N\times N}$ is then formed by the weight $w_{i,j}$ and measures the similarity between nodes. The degree matrix $\mathbf{D}\in\mathbb{R}^{N\times N}$ is a diagonal matrix with each entry the degree (sum of each row of $\mathbf{W}$) of each node.

A graph signal $\mathbf{u}$ is often defined as a discrete signal on the nodes of the graph $\mathbf{u}:\mathcal{V}\to\mathbb{R}$. The discrete signal $\mathbf{u}$ can be regarded as a vector $\mathbf{u}\in\mathbb{R}^N$, where the i-th entry represents the signal value at the i-th node in $\mathcal{V}$. In terms of 2D discrete images, each pixel represents a node, and the pixel intensity stands for the signal value.

2.2 The optimal graph Laplacian regularization algorithm

The OGLR algorithm seeks for a metric space to measure the similarity of image patches [21]. For each pixel location in a 2D image, a vector $\mathbf{v}_i$ of length M is constructed by using a set of exemplar functions $\{\mathrm{f_m}\}_{m=1}^{M}$:

$$\mathbf{v}_i = [\mathrm{f}_1(i), \mathrm{f}_2(i), \ldots, \mathrm{f}_M(i)]. \tag{2}$$

The set of vector $\{\mathbf{v}_i\}_{i=1}^{M}$ is used to build the weighted graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{W})$ with N vertices, where N is the total number of pixels. The determination of the exemplar functions is induced from a continuous graph Laplacian regularizer, described by an anisotropic Dirichlet energy functional $E(u)$:

$$E(u) = \int_{\Omega} \nabla u^T \mathbf{G}^{-1} \nabla u (\sqrt{\det \mathbf{G}})^{2\gamma - 1} \mathrm{ds}, \tag{3}$$

where $s \in \Omega$ is the pixel location. The metric tensor $\mathbf{G} : \Omega \mapsto \mathbb{R}^{2\times 2}$ can be viewed as the structure tensor at s constructed according to the gradients $\{\nabla \mathrm{f}\}_{\mathrm{m}=1}^{\mathrm{M}}$:

$$\mathbf{G} = \sum_{m=1}^{M} \nabla \mathrm{f_m} \nabla \mathrm{f_m}^T \tag{4}$$

An optimal metric tensor $\mathbf{G}^*$ can be estimated by considering the noise model from patch gradients in the MMSE sense:

$$\mathbf{G}^* = \tilde{g}\tilde{g}^T + \alpha_g I, \tag{5}$$

where $\tilde{g}$ is the average gradient of a patch, and the constant α_g is determined by the covariance of the patch. With the estimated $\mathbf{G}^*$, the exemplar functions can be expressed in the following form:

$$\begin{aligned} \mathrm{f}_1^*(i) &= \sqrt{\alpha_g} x_i \\ \mathrm{f}_2^*(i) &= \sqrt{\alpha_g} y_i, \\ \mathrm{f}_3^*(i) &= \frac{1}{L + \sigma_g^2/\sigma_p^2} \sum\nolimits_{l=0}^{L-1} z_l \end{aligned} \tag{6}$$

where (x_i, y_i) are the coordinates of pixel i, and $\{z_l\}_{l=0}^{L}$ is a set of L non-local patches that are similar to z_0. These similar patches are obtained by using the K-Nearest-Neighbour (KNN) algorithm, which seeks L patches with the smallest Euclidean distance from the current patch z_0. Here σ_p is a given constant over the whole noisy image, and σ_g is an estimated variance of the gradient of the patch. f_1^* and f_2^* indicate the spatial relationship between pixels, f_3^* represents the average pixel intensity of a target patch. Note that the coefficient in f_1^*, f_2^* and f_3^* can balance the contributions of the spatial and intensity factors. Hence, the three exemplar functions defined in Eq. (6) can be used to construct the optimal graph edge weight.

2.3 The multi-layer framework

A K-layer tree structure can represent the image hierarchy from fine to coarse, and all the leaf nodes can sum up to the input image. In the multi-layer scheme, the output filtered image u_{out} can be described by a smooth term and several detail terms:

$$u_{out} = \beta_0 u_{smooth} + \beta_1 u_{\mathrm{detail}_1} + \cdots + \beta_K u_{\mathrm{detail}_K}, \tag{7}$$

where $\{\beta_0, \beta_1, \ldots, \beta_K\}$ is a set of coefficients that controls the smoothness and the detail preservation of the output image. More details on the multi-layer scheme can be found in [27, 28].

3 Graph edge weights driven NLMs in a multi-layer framework

Although the OGLR algorithm has excellent filtering performance, it needs numerous iterations to achieve a comparable result. Moreover, it involves a large amount of inverse operation during the denoising process, thus leading to a very high computational cost. Hence in this paper, we would like to take advantage of the edge weights defined in OGLR and apply it to the NLMs algorithm. Then we embed the newly obtained NLMs method into a multi-layer scheme. The multi-layer scheme can decompose input image details from fine to coarse scale, where the fine scale is used to preserve image details and the coarse scale helps smooth the image. The proposed pipeline is shown in Fig. 1.

3.1 NLMs kernel

The NLMs kernel is computed based on the graph edge weights. With the exemplar functions $\{\mathrm{f_m}\}_{m=1}^{M}$, the vector $\mathbf{v}_i$ on node v_i is as follows:

$$\mathbf{v}_i = \left[\sqrt{\alpha_g}x_i, \sqrt{\alpha_g}y_i, 1/(L + \sigma_g^2/\sigma_p^2)\sum_{l=0}^{L-1} z_l\right]. \tag{8}$$

Then the distance ϕ_{ij} in Eq. (1) between node v_i and v_j can be obtained by:

$$\phi_{ij} = \|\mathbf{v}_i - \mathbf{v}_j\|, \tag{9}$$

where $\|\cdot\|$ is the $\mathfrak{L}_2$ norm and ϕ_{ij} determines the weighted affinity matrix $\mathbf{W}$ according to Eq. (1). The diagonal elements of the degree matrix $\mathbf{D}$ is defined as:

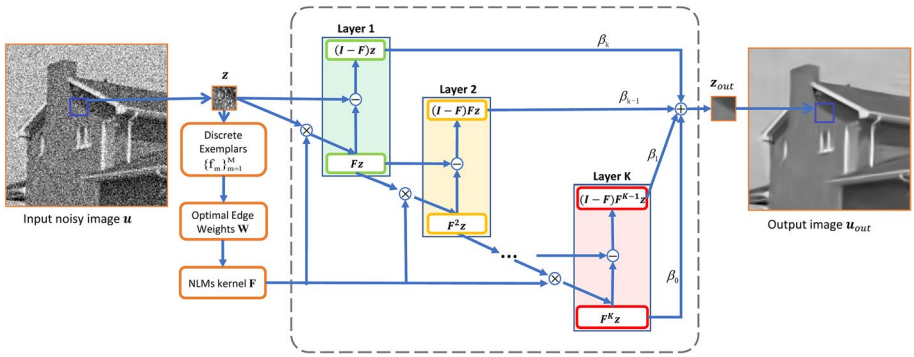

Fig. 1 Flowchart of the proposed algorithm

$$\mathbf{D}_{ii} = \sum_{j} w_{ij}. \tag{10}$$

The NLMs kernel $\mathbf{F}$ is a normalized version of the weight matrix and obtained by the product of $\mathbf{D}^{-1}$ and $\mathbf{W}$:

$$\mathbf{F} = \mathbf{D}^{-1}\mathbf{W} \tag{11}$$

The NLMs kernel $\mathbf{F}$ is similar to the graph-based bilateral filter [12] and the classical NLMs kernel [3]. The difference lies in that $\mathbf{F}$ considers the spatial relationship between pixels and the average intensity of patches. In addition, the relationship and the average intensity are weighted by the gradient estimates, which helps to improve the denoising performance. On the one hand, when the image is polluted by high-level noise, the spatial relationship between pixels dominates the denoising process (like a Gaussian filter). On the other hand, when the signal-noise ratio (SNR) is high, the average intensity plays a more critical role.

It is worthy to note that the OGLR algorithm denoises the target patch z_0 by calculating the inverse of the Laplace operator, i.e., $z^* = (\mathbf{I} + \tau\mathbf{L})^{-1}z_0$. Although $\mathbf{L}$ is of small size, the inverse operation still costs a lot of time. On the contrary, our NLMs kernel works forward, which avoids the inverse operation as done in the OGLR algorithm (except for the inverse of the diagonal matrix D, a linear operation). Hence, our method works much faster than the OGLR method.

3.2 Determine the coefficients with least square

The set of coefficients $\{\beta_k\}$ in the multi-layer scheme plays a significant role in achieving good denoising performance. In this paper, instead of using parameters according to the s-curve functions proposed in [28], we regard the determination of $\{\beta_k\}$ as a regression problem and apply the least square algorithm to solve it. Our cost function is as follows:

$$C = \min_{\{\beta_0,\ldots\beta_K\}} \sum_{p=1}^{P} \left\| \left(\beta_0 \mathbf{F}^K + \sum_{k=1}^{K} \beta_k (\mathbf{I} - \mathbf{F}) \right) \mathbf{F}^{K-k} z_p - z_{0p} \right\|_2^2, \tag{12}$$

where K is the number of layers, P is the total number of training images, z_p represents the p-th noisy image patch, z_{0p} stands for the p-th noise-free image patch. The aim of (12) is to find an appropriate series of $\{\beta_k\}$ that work on different filters to minimize the difference between the noisy and clean image patches.

Note that during the training process, we distinguish the images with different noise levels. In other words, each noise level will be assigned with a set of optimal coefficients. For each noise level, when the training process is finished, we will estimate the noise variance according to the newly-obtained $\{\beta_k\}$. If the estimated noise is higher than a given threshold σ_{th}, it is encouraged to train $\{\beta_k\}$ again with the newly-obtained $\{\beta_k\}$.

Additionally, the number of layers K is also an important parameter. Details will be discussed in Sects. 3.3 and 4.2.

3.3 NLMs with K residual compensation

The NLMs filter can be embedded into the multi-layer scheme and the output filtered image is with one smooth term and K residuals:

$$u_{out} = \beta_0 \mathbf{F}^K u + \beta_1 (\mathbf{I} - \mathbf{F}) \mathbf{F}^{K-1} u + ... + \beta_{K-1} (\mathbf{I} - \mathbf{F}) \mathbf{F} u + \beta_K (\mathbf{I} - \mathbf{F}) u. \tag{13}$$

Since $\mathbf{F}$ is the normalized affinity matrix, it can act as a low-pass filter according to the graph Fourier transform theory. $(\mathbf{I} - \mathbf{F})$ is the normalized Laplacian, and it can function as a high-pass filter [5]. $\mathbf{F}^K u$ stands for the smooth term, which is obtained by the cascade of K low-pass filters $\mathbf{F}$. The residual K terms are the corresponding detail terms. When $K = 1$, Eq.(13) degenerates to:

$$u_{out} = \beta_0 \mathbf{F} u + \beta_1 (\mathbf{I} - \mathbf{F}) u,$$

where the filtered image is composed of one smooth term $\mathbf{F}u$ and one residual detail term $(\mathbf{I} - \mathbf{F})u$. Thus, when K increases, the smoother u_{out} will be. The value of K can not be too large or too small. Too few layers may lead to an incomplete representation of the image, which can not remove the noise effectively, i.e., some details are not restored, or the homogeneous part of the filtered image is not smooth enough *etc.* However, too many layers would result in a large computation work, which consumes a lot of time, with only a slight performance improvement. The choice of K will be discussed in Sect. 4.2.

With the learned coefficients $\{\beta_k\}$ and the number of layers K, the proposed method is summarized in Algorithm 1. In addition, the flowchart of the proposed graph-based NLMs with multi-layer residual compensation is shown in Fig. 1, where a noisy image with noise variance $\sigma = 50$ is used as an example.

Algorithm 1: Graph-based NLMs with Multi-Layer Residual Compensation

Input: Noisy image $u(x, y)$, number of layers K, coefficients $\{\beta_k\}$ noise variance σ;
Output: Filtered image u_{out};

foreach Noisy patch z in u **do**
1. Find L similar patches of z, estimate the variance σ_g of the noisy gradient from $\{z_l\}_{l=0}^{L}$;
2: Compute the exemplar functions $\mathrm{f_m}$ and build the graph with the optimal edge weights $\mathbf{W}$ accordingly;
3: Determine the NLMs kernel $\mathbf{F}$ based on $\mathbf{W}$;
4: Denoise current patch z according to Eq.(13).
endfor
Integration of the denoised patches z to acquire u_{out}
return u_{out}

4 Experimentation

4.1 Experimental setup

We testify the effectiveness of the proposed method both on natural images and depth images. Additive white Gaussian noise (AWGN) is added to these images, with standard deviations σ ranging from 10 to 50. According to different noise variances σ, the patch size in our experiment ranges from 10 to 22, the and step size N_S is from 2 to 6. In the implementation, the normalization parameter γ in Eq. (3) was empirically set to be 0.6 for the natural images and $\gamma = 1$ for the depth images. The constant σ_p in Eq. (6) is set to be 10^6 and the patch cluster size L is from 5 to 50. The noise variance threshold mentioned in Sec.3.2 is $\sigma_{th} = 5$.

The test images are from public dataset such as the BSDS500 dataset [1] and the Middlebury Stereo Datasets [23].

We compare our method with the original NLMs [3], OGLR [21] and two other state-of-the-art methods, i.e., Block-Matching 3D (BM3D) [8] and the ADNet method [30]. The peak signal noise ratio (PSNR) and the structural similarity (SSIM) are used to evaluate the performance of these methods.

4.2 Determination of number of layers *K*

To find out the most appropriate number of layers K, we test six different images. Five levels of noise are tested separately, i.e., $\sigma = 10, 20, 30, 40, 50$. The average PSNR and SSIM of test images under different noise levels are computed with different K. In our experiments, K ranges from 1 to 6. The maximum of K is set to be 6 because when $K > 6$, the computation of the power of matrix costs a lot of time, which is contrary to our motivation.

Figure 2 shows the PSNR (a) and SSIM (b) results according to K. We can see from (a) that when $K \geq 4$, the PSNR converges to a fixed value for all five noise levels. Given SSIM (b), the more layers there are, the higher the SSIM, but when $K \geq 4$, it does not improve significantly. Hence, to balance the PSNR, SSIM, and time cost, we make a compromise by setting $K = 4$ in our algorithm according to Fig. 2.

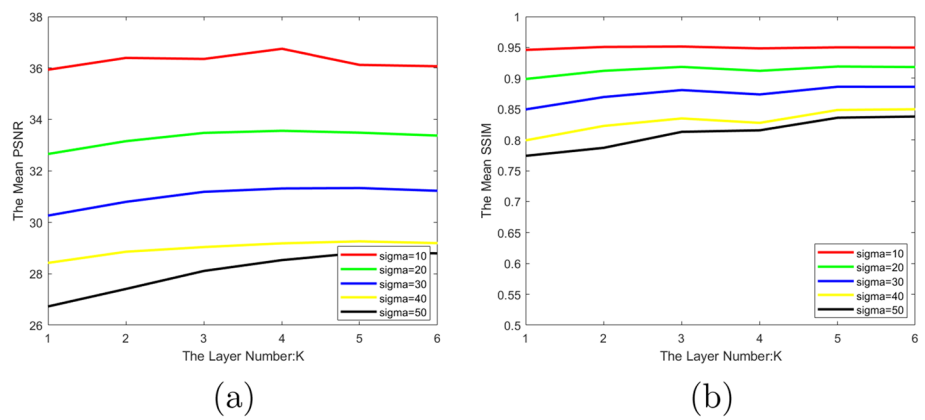

Fig. 2 From above to bottom: **a** the average PSNR (dB) of different number of layers, **b** the average SSIM of different number of layers

4.3 Determination of $\{\beta_k\}$

In the process of training the coefficient set $\{\beta_k\}$ in Eq. (13), three depth images(*cloth*, *Aloe* and *flowerpot*) and three natural images (*barbara*, *chips* and *man*) are used as the training data, as shown in Fig. 3. Each image is divided into 3000 to 6000 image patches under different noise levels with varying patch sizes.

The learned sets $\{\beta_k\}$ are shown in Table 1 and 2. Both the two tables indicate that the smooth term $\mathbf{F}^K u$ of Eq. (13) plays the major role in the denoising process. Additionally, although the coefficients of the residual terms are small values, even negative values, they also play important roles in retaining detailed information and removing noise.

5 Results and discussion

Figures 4, 5, 6, and 7 depict the denoising performance of the five methods on four depth images (*wood*, *bowling*, *lampshade*, and *teddy*) with a noise variance $\sigma = 50$, the difference between the original depth images and the filtered images, and the corresponding zoomed parts. Figures 8, 9, and 10 show the denoising results of the five methods on three natural images (*bird*, *house* and *jar*) with noise variance $\sigma = 50$ respectively. From left to right are: the original image, the noisy image, the results obtained by OGLR, BM3D, ADNet and the proposed method.

(a) (b) (c) (d) (e) (f)

Fig. 3 The images in the process of training the coefficient set $\{\beta_k\}$. Depth images: **a** *cloth*, **b** *Aloe*, **c** *flowerpot*; natural images: **d** *barbara*, **e** *chips*, **f** *man*

Table 1 The learned $\{\beta_k\}$ for depth images with different levels of noise

	β_0	β_1	β_2	β_3	β_4
$\sigma < 5$	1.0223	0.0002	0.0001	0.0001	0.0021
$\sigma = 10$	0.9992	0.0000	− 0.0005	− 0.0001	0.0002
$\sigma = 20$	0.9976	− 0.0003	− 0.0004	− 0.0004	− 0.0002
$\sigma = 30$	0.9766	− 0.0006	− 0.0007	− 0.0009	− 0.0001
$\sigma = 40$	0.9885	− 0.0012	− 0.0013	− 0.0019	0.0011
$\sigma = 50$	0.9862	− 0.0021	− 0.0025	− 0.0032	− 0.0016

Table 2 The learned $\{\beta_k\}$ for real natural images with different levels of noise

	β_0	β_1	β_2	β_3	β_4
$\sigma < 5$	1.0058	0.0002	0.0003	− 0.0003	0.0052
$\sigma = 10$	0.9996	0.0003	− 0.0001	− 0.0003	0.0069
$\sigma = 20$	0.9946	0.0001	0.0001	0.0001	0.0046
$\sigma = 30$	0.9940	0.0000	0.0001	0.0001	0.0070
$\sigma = 40$	0.9955	− 0.0004	− 0.0003	− 0.0008	0.0153
$\sigma = 50$	0.9903	− 0.0012	− 0.0016	− 0.0016	0.0085

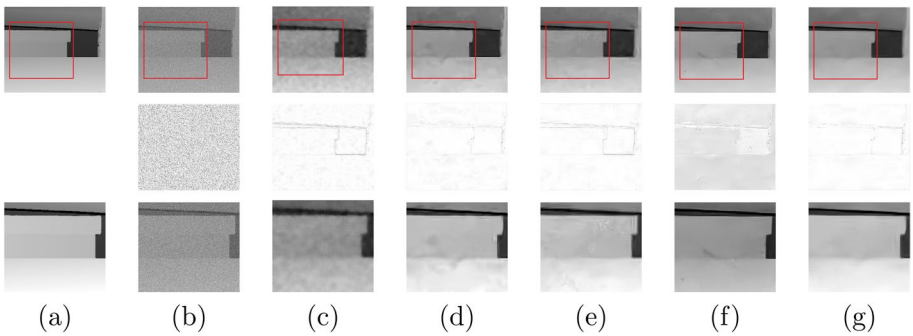

Fig. 4 Denoising results of depth image (*wood*) with the noise variance $\sigma = 50$ [top], the difference between the original depth images and the filtered images [middle], and zoom-in image [bottom]. From the left (**a**) to right (**g**) are: the original image, the noisy image, the results obtained by NLM, BM3D, OGLR, ADnet and the proposed method respectively

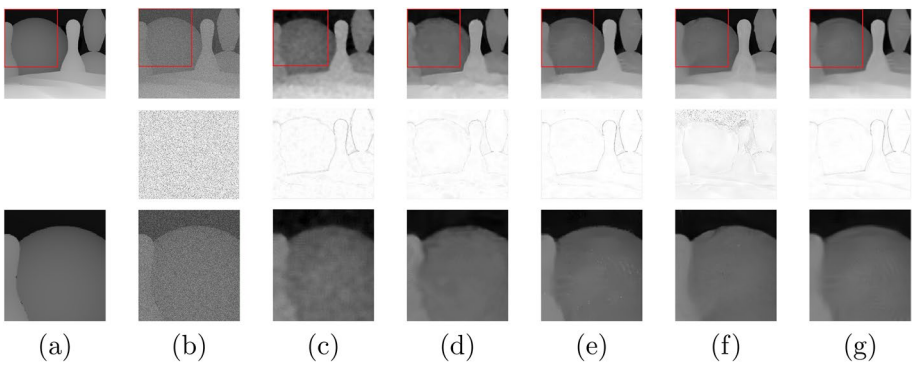

Fig. 5 Denoising results of depth image (*bowling*) with the noise variance $\sigma = 50$ [top], the difference between the original depth images and the filtered images [middle], and zoom-in image [bottom]. From the left (**a**) to right (**g**) are: the original image, the noisy image, the results obtained by NLM, BM3D, OGLR, ADnet and the proposed method respectively

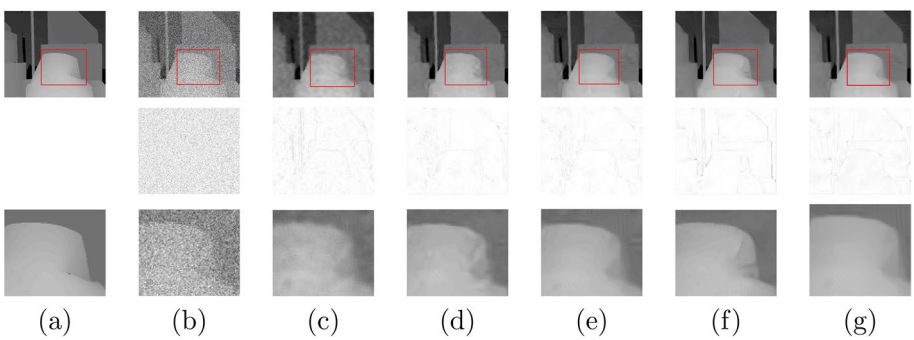

Fig. 6 Denoising results of depth image (*lampshade*) with the noise variance $\sigma = 50$ [top], the difference between the original depth images and the filtered images [middle], and zoom-in image [bottom]. From the left (**a**) to right (**g**) are: the original image, the noisy image, the results obtained by NLM, BM3D, OGLR, ADnet and the proposed method respectively

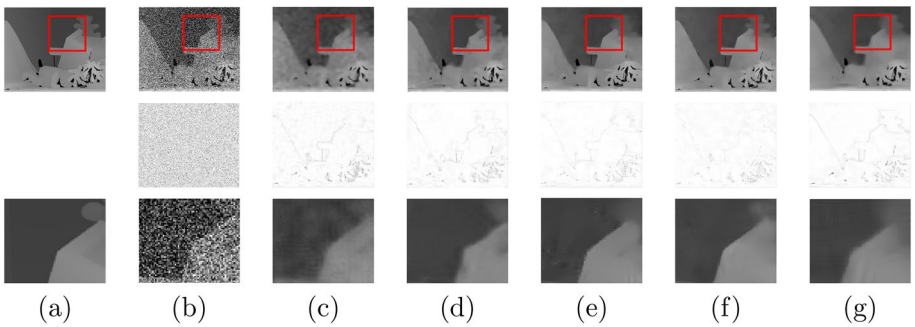

Fig. 7 Denoising results of depth image (*teddy*) with the noise variance $\sigma = 50$ [top], the difference between the original depth images and the filtered images [middle], and zoom-in image [bottom]. From the left (**a**) to right (**g**) are: the original image, the noisy image, the results obtained by NLM, BM3D, OGLR, ADnet and the proposed method respectively

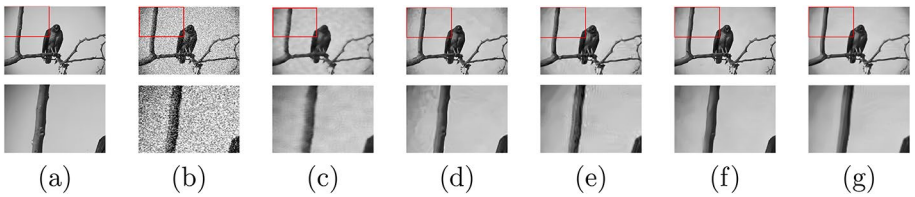

Fig. 8 Denoising results of natural image (*bird*) with the noise variance $\sigma = 50$ [top] and zoom-in image [bottom]. From the left (**a**) to right (**g**) are: the orig inal image, the noisy image, the results obtained by NLM, BM3D, OGLR, ADnet and the proposed method respectively

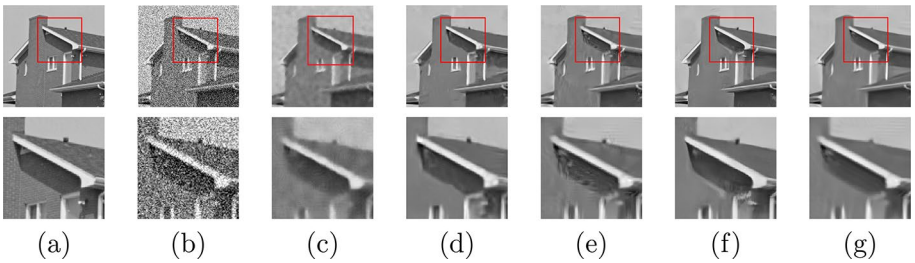

Fig. 9 Denoising results of natural image (*house*) with the noise variance $\sigma = 50$ [top] and zoom-in image [bottom]. From the left (**a**) to right (**g**) are: the original image, the noisy image, the results obtained by NLM, BM3D, OGLR, ADnet and the proposed method respectively

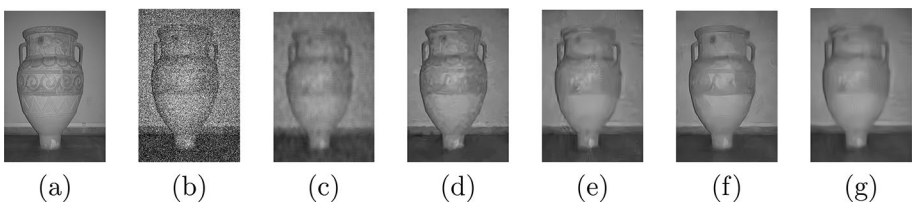

Fig. 10 Denoising results of natural image (*jar*) with the noise variance $\sigma = 50$ [top]. From the left (**a**) to right (**g**) are: the original image, the noisy image, the results obtained by NLM, BM3D, OGLR, ADnet and the proposed method respectively

For the image *wood*, the horizontal line in the center of the image is seriously blurred by OGLR and BM3D. Moreover, the homogeneous parts are still corrupted and not well restored. ADNet can preserve edges very well visually. However, it generates some undesirable parts, such as the black point in the lower-left corner and the black segment in the center. In our case, although the edges are not preserved as well as ADNet, the homogeneous parts are well smoothed. The middle row shows the figures of differences between the original image and filtered images. Dark contours and areas indicate that there are significant differences. From the difference figures, we can see that our result is more similar to the original clean image. In terms of both visual performance and the indexes, our method provides the best denoising result.

For the image *bowling*, the PSNR and SSIM of the proposed method are superior to the other three methods. The edge between the bowling ball and pin is blurred, even distorted by BM3D. ADNet generates a deformation on the edge of the ball. The deformation may be due to that the training data does not include images with this kind of data and shape. Our method and OGLR provide better results, while our result is smoother in the homogeneous regions. Additionally, the figures of differences demonstrate that our method has excellent denoising performance and preserves the brightness very well.

The images *lampshade* and *teddy* are more complex than the above two images, with more details and weak edges, shown in Figs. 6 and 7. The results show that our method can restore the image very well in smooth areas, but can not preserve the sharp corners, *e.g.*, the corner of the zoomed parts. In addition, our method generates some artifacts in the homogeneous areas. Actually, this phenomenon exists in the NLM theory-based methods, including the NLM, BM3D, OGLR. When the SNR is low, i.e., the noise is strong, some neighboring pixels in a patch may be considered as line structures. These structures will be enhanced or preserved during denoising, thus producing the artifacts.

Figures 8 and 9 show the denoising results of two natural images *bird* and *house* under noisy case $\sigma = 50$. It is clear that the proposed method achieves satisfactory performance in the homogeneous region of the image, like the sky in *bird* and the shadow of the roof in *house*. From the enlarged sub-images in Fig. 9, one can see that BM3D, OGLR and ADNet generate some undesirable artifacts and destroy the edge of shadow, while our method provides smoother results with fewer artifacts. Figure 10 shows the results on *jar*, the carved patterns on the jar can be barely seen after denoising. This hints that our method is not that effective in maintaining the details of texture-rich images.

Figure 11 display the examples on two color images *tape* and *pepper* under noisy case $\sigma = 50$. The color image are denoised channel by channel. Visually speaking, the denoising results of our method is competitive with the other methods

Tables 3 and 4 illustrate the PSNR and SSIM of the proposed method and four other state-of-the-art methods on several depth images and real natural images. The highest indexes are in bold, the second-best are underlined. From the results, we can see that our proposed method is comparable with the state-of-the-art methods. In addition, it outperforms the OGLR method in nearly all cases, especially with a large noise variance σ. Furthermore, when σ is large, the performance of our method becomes more competitive. In addition, our method performs better for the piece-wise depth image compared to the performance on real natural images. However, when dealing with texture-rich images such as *jar* and *flower*, our result is not as good as ADNet.

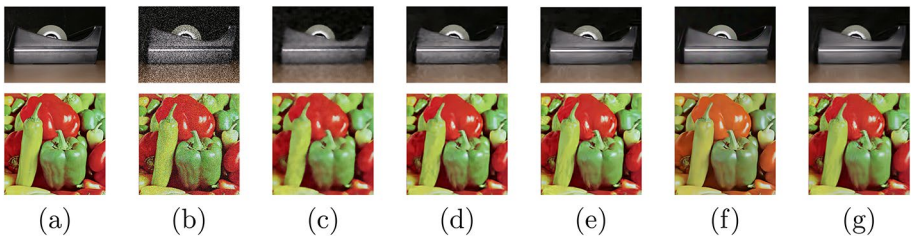

Fig. 11 Denoising results of color images (*tape* and *Pepper*) with the noise variance $\sigma = 50$. From the left (**a**) to right (**g**) are: the original image, the noisy image, the results obtained by NLM, BM3D, OGLR, ADnet and the proposed method respectively

Table 3 Image denoising on depth images with NLM ,BM3D, OGLR, ADNet and our method: peformance comparisons in PSNR (Left, in dB) and SSIM (Right)

Images	noise	NLM	BM3D	OGLR	ADNet	Our Method
teddy	$\sigma = 10$	37.04dB 0.9725	40.10dB 0.9818	39.98dB 0.9833	**41.98dB 0.9878**	38.24dB 0.9673
	$\sigma = 20$	33.60dB 0.9336	35.94dB 0.9674	35.94dB 0.9644	**37.74dB 0.9779**	34.69dB 0.9644
	$\sigma = 30$	30.89dB 0.8842	33.16dB 0.9481	33.49dB 0.9441	**35.07dB 0.9669**	33.02dB 0.9486
	$\sigma = 40$	29.04dB 0.8299	31.32dB 0.9279	31.78dB 0.9277	**33.15dB 0.9539**	30.91dB 0.9288
	$\sigma = 50$	27.71dB 0.7741	29.73dB 0.9190	30.46dB 0.9049	**31.50dB 0.9404**	30.38dB 0.9125
wood	$\sigma = 10$	43.64dB 0.9870	42.21dB 0.9889	**44.45dB** 0.9882	43.46dB **0.9924**	42.50dB 0.9701
	$\sigma = 20$	38.39dB 0.9559	38.09dB 0.9727	40.56dB 0.9761	39.79dB **0.9862**	**41.09dB** 0.9839
	$\sigma = 30$	35.17dB 0.9137	35.94dB 0.9573	38.15dB 0.9600	37.82dB **0.9792**	**38.91dB** 0.9770
	$\sigma = 40$	32.83dB 0.8620	34.46dB 0.9420	36.57dB 0.9478	36.58dB **0.9746**	**37.47dB** 0.9656
	$\sigma = 50$	31.06dB 0.8159	33.28dB 0.9375	34.55dB 0.9184	35.41dB **0.9664**	**36.07dB** 0.9512
Sawtooth	$\sigma = 10$	43.77dB 0.9866	44.62dB 0.9884	45.35dB 0.9911	**45.61dB 0.9912**	43.40dB 0.9731
	$\sigma = 20$	38.12dB 0.9513	41.16dB 0.9787	40.96dB 0.9779	**42.72dB 0.9873**	41.11dB 0.9865
	$\sigma = 30$	34.78dB 0.9032	38.72dB 0.9655	38.34dB 0.9630	**40.41dB 0.9802**	39.56dB 0.9762
	$\sigma = 40$	32.69dB 0.8484	36.84dB 0.9497	36.51dB 0.9526	**38.84dB 0.9747**	37.29dB 0.9665
	$\sigma = 50$	31.19dB 0.7912	35.87dB 0.9500	34.94dB 0.9334	**37.34dB 0.9654**	35.84dB 0.9510
Tsukuba	$\sigma = 10$	39.71dB 0.9724	41.52dB 0.9798	**41.64dB 0.9850**	41.41dB 0.9813	40.67dB 0.9701
	$\sigma = 20$	35.53dB 0.9308	37.38dB 0.9577	37.56dB 0.9626	**37.76dB 0.9667**	37.28dB 0.9657
	$\sigma = 30$	32.37dB 0.8743	34.81dB 0.9319	34.91dB 0.9403	**35.58dB 0.9486**	35.23dB 0.9468
	$\sigma = 40$	30.05dB 0.8110	33.05dB 0.9053	32.99dB 0.9245	**34.00dB** 0.9244	33.41dB **0.9313**
	$\sigma = 50$	28.36dB 0.7497	31.67dB 0.8915	31.44dB 0.9003	**32.78dB 0.9157**	32.57dB 0.9133
Books	$\sigma = 10$	40.28dB 0.9727	40.89dB 0.9811	**42.37dB** 0.9828	41.83dB **0.9838**	41.69dB 0.9827
	$\sigma = 20$	36.37dB 0.9351	35.10dB 0.9519	**38.26dB** 0.9625	37.01dB 0.9609	37.35dB **0.9666**
	$\sigma = 30$	33.89dB 0.8887	32.61dB 0.9235	**35.88dB** 0.9420	34.45dB 0.9461	35.62dB **0.9491**
	$\sigma = 40$	32.18dB 0.8373	31.22dB 0.8970	34.10dB 0.9256	32.44dB 0.9260	**34.28dB 0.9334**
	$\sigma = 50$	30.86dB 0.7834	29.76dB 0.8812	32.86dB 0.9050	31.11dB 0.9147	**33.31dB 0.9154**
Bowling	$\sigma = 10$	**43.18dB**0.9850	41.62dB 0.9850	42.70dB 0.9867	43.09dB **0.9899**	42.12dB 0.9725
	$\sigma = 20$	38.65dB 0.9542	37.72dB 0.9682	38.84dB 0.9723	**39.48dB** 0.9796	39.32dB **0.9816**
	$\sigma = 30$	35.62dB 0.9126	35.30dB 0.9484	36.45dB 0.9573	37.46dB 0.9699	**38.22dB 0.9732**
	$\sigma = 40$	33.32dB 0.8622	33.60dB 0.9379	34.73dB 0.9481	36.03dB 0.9624	**36.27dB 0.9654**
	$\sigma = 50$	31.54dB 0.8069	32.18dB 0.9193	33.70dB 0.9330	34.37dB 0.9466	**35.56dB 0.9531**

Bold and underline to mark the best and the second best results for each quality index, respectively

The above experiments imply that our method is good at denoising piece-wise smooth images. To further testify this conclusion, we test our method on the Middlebury Stereo Datasets 2006 [13]. Table 5 summarizes a mean PSNR and SSIM of

Table 4 Image denoising on nature images with NLM ,BM3D, OGLR, ADNet and our method: performance comparisons in PSNR (Left, in dB) and SSIM (Right)

Images	noise	NLM	BM3D	OGLR	ADNet	Our method
house	$\sigma = 10$	37.55dB 0.9504	36.71dB 0.9212	**38.88dB 0.9622**	36.57dB 0.9077	38.50dB 0.9533
	$\sigma = 20$	33.86dB 0.9146	33.77dB 0.8721	**35.87dB** 0.9405	34.12dB 0.8713	35.68dB **0.9448**
	$\sigma = 30$	31.01dB 0.8659	32.09dB 0.8473	**33.86dB** 0.9183	32.62dB 0.8556	33.79dB **0.9279**
	$\sigma = 40$	29.03dB 0.8113	30.65dB 0.8249	**32.49dB** 0.9024	31.26dB 0.8387	31.33dB **0.9085**
	$\sigma = 50$	27.61dB 0.7559	29.69dB 0.8116	30.67dB 0.8631	30.28dB 0.8230	**31.13dB 0.8969**
church	$\sigma = 10$	37.70dB 0.9600	39.53dB 0.9670	39.59dB **0.9744**	**40.38dB** 0.9710	39.18dB 0.9629
	$\sigma = 20$	33.35dB 0.9151	35.99dB 0.9455	36.04dB 0.9527	**37.23dB 0.9580**	35.68dB 0.9564
	$\sigma = 30$	30.54dB 0.8591	33.92dB 0.9254	33.71dB 0.9292	**35.06dB 0.9413**	33.59dB 0.9378
	$\sigma = 40$	28.84dB 0.8021	32.42dB 0.9056	32.13dB 0.9048	**33.64dB 0.9398**	30.80dB 0.9101
	$\sigma = 50$	27.63dB 0.7463	31.33dB 0.8974	30.60dB 0.8636	**32.37dB 0.9142**	30.66dB 0.8945
flower	$\sigma = 10$	35.69dB 0.9469	38.15dB 0.9667	37.77dB 0.9655	**38.70dB 0.9712**	37.78dB 0.9620
	$\sigma = 20$	32.40dB 0.8986	34.29dB 0.9314	34.33dB **0.9342**	**35.22dB** 0.9476	34.15dB 0.9362
	$\sigma = 30$	30.26dB 0.8497	32.19dB 0.8988	31.95dB **0.8998**	**33.17dB** 0.9225	31.94dB 0.9049
	$\sigma = 40$	28.90dB 0.8033	30.67dB 0.8679	30.51dB **0.8713**	**31.78dB** 0.9046	29.73dB 0.8661
	$\sigma = 50$	27.90dB 0.7575	29.72dB 0.8529	29.09dB **0.8231**	**30.44dB** 0.8813	29.37dB 0.8544
jar	$\sigma = 10$	35.71dB 0.8954	**38.63dB 0.9459**	37.91dB 0.9354	37.52dB 0.9349	38.01dB 0.9333
	$\sigma = 20$	33.33dB 0.8471	**35.26dB 0.8986**	34.96dB 0.8911	34.58dB 0.8887	34.93dB 0.8884
	$\sigma = 30$	31.63dB 0.7992	**33.41dB 0.8637**	33.29dB 0.8598	32.96dB 0.8506	33.40dB 0.8615
	$\sigma = 40$	30.43dB 0.7534	**32.04dB** 0.8350	32.09dB 0.8375	32.02dB 0.8222	32.02dB **0.8403**
	$\sigma = 50$	29.48 0.7083	**31.17dB** 0.8191	31.00dB 0.8128	31.13dB 0.8002	31.08dB **0.8204**
bird	$\sigma = 10$	35.80dB 0.9735	38.28dB **0.9829**	37.04dB 0.9796	**38.55dB** 0.9754	36.41dB 0.9679
	$\sigma = 20$	32.42dB 0.9385	34.30dB **0.9651**	33.62dB 0.9621	**35.20dB** 0.9615	32.79dB 0.9635
	$\sigma = 30$	29.66dB 0.8928	31.99dB 0.9499	31.58dB 0.9421	**33.21dB 0.9506**	31.38dB 0.9503
	$\sigma = 40$	27.66dB 0.8407	**31.63dB 0.9377**	30.01dB 0.9201	**31.63dB 0.9377**	28.47dB 0.9261
	$\sigma = 50$	26.16dB 0.7851	28.95dB 0.9101	28.62dB 0.8830	**30.62dB** 0.9189	29.02dB **0.9213**

Bold and underline to mark the best and the second best results for each quality index, respectively

Table 5 The results of image denoising on depth dataset

noise	NLM	BM3D	OGLR	ADNet	Our Method
$\sigma = 10$	41.96dB 0.9797	**45.20dB 0.9887**	44.40dB 0.9859	43.63dB 0.9864	43.36dB 0.9779
$\sigma = 20$	37.42dB 0.9742	**40.88dB 0.9749**	40.47dB 0.9705	40.19dB 0.9746	39.46dB 0.9601
$\sigma = 30$	34.53dB 0.8983	**38.13dB** 0.9581	38.02dB 0.9522	37.80dB **0.9598**	37.67dB 0.9543
$\sigma = 40$	32.53dB 0.8463	36.08dB 0.9389	36.36dB 0.9368	36.17dB 0.9487	**36.62dB 0.9581**
$\sigma = 50$	31.11dB 0.7915	**35.16dB** 0.9368	34.40dB 0.9046	34.84dB 0.9379	34.94dB **0.9445**

Bold and underline to mark the best and the second best results for each quality index, respectively

10 depth images obtained by the five methods. The results show that our method is effective at denoising the depth images, i.e., piece-wise smooth images, and our method outperforms the other methods when the noise level $\sigma > 30$. This is due to that we use the multi-layer framework: the term $\mathbf{F}^K\mathbf{u}$ is obtained after K filtering, which results in a very smooth term nearly with no noise. The residual terms function as supplements, which helps to restore some details from the noise. Furthermore, our method takes considerably less time to operate than the OGLR method. For instance, our method takes around 30 seconds to process an image with a size of 500*300 pixels

and a noise level of sigma=10, whereas the OGLR takes about 90 seconds. However, as compared to other approaches such as BM3D and ADnet, the graph-based methods take longer, which is a common downside of the graph-based method.

6 Conclusion

In this paper, we propose a graph-based NLMs algorithm for image denoising. The edge weights defined in the OGLR algorithm are applied as the NLMs kernel. A multi-layer residual compensation strategy is then used to recover the details. The coefficients of the smooth term and the residual terms of the multi-layer representation are learned according to the least mean square method. We testify the effectiveness of our method both on natural images and depth images. Our proposed method outperforms the original OGLR method in PSNR/SSIM/time cost. Compared with the other state-of-the-art methods, including the classical NLMs, BM3D and the AD-Net, our proposed method provides comparable or better results. Especially, our method has excellent denoising performance on the piecewise smooth images when the noise level is high.

Abbreviations

NLMs: Non-local means; TV: Total variation; BM3D: 3D transform-domain filter; PCA: Principal component analysis; GSP: Graph signal processing; OGLR: The optimal graph Laplacian regularization; AWGN: Additive white Gaussian noise; PSNR: The peak signal noise ratio; SSIM: The structural similarity.

Acknowledgements

Not applicable.

Authors' contributions

Conceptualization, methodology, writing-original draft, FY; software, validation, XC; supervision, funding acquisition, LC. All authors approved the final, submitted version of the manuscript.

Funding

This research was funded by National Natural Science Foundation of China (61625305).

Availability and data materials

The data that support the findings of this study are available on request from the corresponding author F.Y.

Declarations

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Received: 15 March 2021 Accepted: 16 September 2021
Published online: 28 September 2021

References

1. P. Arbelaez, M. Maire, C. Fowlkes, J. Malik, Contour detection and hierarchical image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **33**(5), 898–916 (2010)
2. T. Bouwmans, S. Javed, H. Zhang, Z. Lin, R. Otazo, On the applications of robust PCA in image and video processing. Proc. IEEE **106**(8), 1427–1457 (2018)
3. A. Buades, B. Coll, J. Morel, A non-local algorithm for image denoising. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2, pp. 60–65 (2005)
4. J. Chen, J. Chen, H. Chao, Y. Ming, Image blind denoising with generative adversarial network based noise modeling. in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2018)
5. G. Cheung, E. Magli, Y. Tanaka, M. Ng, Graph spectral image processing. Proc. IEEE **106**(5), 907–930 (2018)
6. D. Cho, T.D. Bui, Multivariate statistical modeling for image denoising using wavelet transforms. Signal Process.: Image Commun. **20**(1), 77–89 (2005)

7. C. Couprie, L. Grady, L. Najman, J.-C. Pesquet, H. Talbot, Dual constrained TV-based regularization on graphs. SIAM J. Image Sci. **6**(3), 1246–1273 (2013)
8. K. Dabov, A. Foi, V. Katkovnik, K. Egiazarian, Image denoising by sparse 3-D transform-domain collaborative filtering. IEEE Trans. Image Process. **16**(8), 2080–2095 (2007)
9. W. Dong, X. Li, L. Zhang, G. Shi, Sparsity-based image denoising via dictionary learning and structural clustering. in *CVPR 2011* (IEEE, 2011), p. 457–464 (2011)
10. K. Egiazarian, A. Foi, V. Katkovnik, Compressed sensing image reconstruction via recursive spatially adaptive filtering. in *2007 IEEE International Conference on Image Processing*, vol. 1 (IEEE, 2007). p. I–549
11. M. Elad, On the origin of the bilateral filter and ways to improve it. IEEE Trans. Image Process. **11**(10), 1141–1151 (2002)
12. A. Gadde, S.K. Narang, A. Ortega, Bilateral filter: graph spectral interpretation and extensions. in *2013 IEEE International Conference on Image Processing* (IEEE, 2013). p. 1222–1226
13. H. Hirschmüller, D. Scharstein, Evaluation of cost functions for stereo matching. in *IEEE Conference on Computer Vision and Pattern Recognition* (2007)
14. A. Hyvärinen, P. Hoyer, E. Oja, Image denoising by sparse code shrinkage. in *Intelligent Signal Processing* (Citeseer, 1999)
15. A. Kheradmand, P. Milanfar, A general framework for regularized, similarity-based image restoration. IEEE Trans. Image Process. **23**(12), 5136–5151 (2014)
16. J. Liang, R. Liu, Stacked denoising autoencoder and dropout together to prevent overfitting in deep neural network, in *2015 8th International Congress on Image and Signal Processing (CISP)* (2015)
17. C. Louchet, L. Moisan, Total variation as a local filter. SIAM J. Image Sci. **4**(2), 651–694 (2011)
18. F.G. Meyer, X. Shen, Perturbation of the eigenvectors of the graph Laplacian: application to image denoising. Appl. Comput. Harmon. Anal. **36**(2), 326–334 (2014)
19. P. Milanfar, A tour of modern image filtering: New insights and methods, both practical and theoretical. IEEE Signal Process. Mag. **30**(1), 106–128 (2012)
20. M. Onuki, S. Ono, M. Yamagishi, Y. Tanaka, Graph signal denoising via trilateral filter on graph spectral domain. IEEE Trans. Signal Inform. Process. Over Netw. **2**(2), 137–148 (2016)
21. J. Pang, G. Cheung, Graph Laplacian regularization for image denoising: analysis in the continuous domain. IEEE Trans. Image Process. **26**(4), 1770–1785 (2017)
22. J.-M. Park, W.-J. Song, W. Pearlman, Speckle filtering of SAR images based on adaptive windowing. IEE Proc.-Vis. Image Signal Process. **146**(4), 191–197 (1999)
23. D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, P. Westling, High-resolution stereo datasets with subpixel-accurate ground truth, in *German conference on pattern recognition* (Springer, 2014). p. 31–42
24. D.I. Shuman, S.K. Narang, P. Frossard, A. Ortega, P. Vandergheynst, The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Process. Mag. **30**(3), 83–98 (2013)
25. J.-L. Starck, E.J. Candès, D.L. Donoho, The curvelet transform for image denoising. IEEE Trans. Image Process. **11**(6), 670–684 (2002)
26. C. Sutour, C.A. Deledalle, J.F. Aujol, Adaptive regularization of the NL-means: application to image and video denoising. IEEE Trans. Image Process. **23**(8), 3506–3521 (2014)
27. H. Talebi, P. Milanfar, Nonlocal image editing. IEEE Trans. Image Process. **23**(10), 4460–4473 (2014)
28. H. Talebi, P. Milanfar, Fast multilayer Laplacian enhancement. IEEE Trans. Comput. Imaging **2**(4), 496–509 (2016)
29. G. Taubin, A signal processing approach to fair surface design, in *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques* (1995). p. 351–358
30. C. Tian, X. Yong, Z. Li, W. Zuo, L. Fei, H. Liu, Attention-guided CNN for image denoising. Neural Netw. **124**, 177–129 (2020)
31. C. Tian, X. Yong, W. Zuo, Image denoising using deep CNN with batch renormalization. Neural Netw. **121**, 461–473 (2020)
32. G. Vasile, E. Trouve, J.S. Lee, V. Buzuloiu, Intensity-driven adaptive-neighborhood technique for polarimetric and interferometric SAR parameters estimation. IEEE Trans. Geosci. Remote Sens. **44**(6), 1609–1621 (2006)
33. J. Zeng, G. Cheung, M. Ng, J. Pang, C. Yang, 3D point cloud denoising using graph Laplacian regularization of a low dimensional manifold model. IEEE Trans. Image Process. **29**, 3474–3489 (2019)
34. K. Zhang, W. Zuo, Y. Chen, D. Meng, L. Zhang, Beyond a Gaussian denoiser: residual learning of deep CNN for image denoising. IEEE Trans. Image Process. **26**(7), 3142–3155 (2016)

Publisher's Note