Оптический журнал

Иконика

УДК 535; 617.7; 628.9

Алгоритм снижения шумов изображения на основе сглаживания, использующего ориентированные модели

© 2020 г. Xiaoming Zhao, Yashuo Bai, Xin Liu, Miao Gao, Kun Cheng, Shengcun Ma, Lei Dong

Целью работы является сохранение резкости границ в процессе обработки изображений, снижающем уровень шумов, для улучшения комфортности восприятия картины без уменьшения заметности границ и деталей. В предлагаемом алгоритме при выделении границ направление их ориентации оценивается с использованием ориентированных образцов-моделей. Результаты свертки используются для управления значениями коэффициентов в соответствующих шумоподавляющих фильтрах. Использование такой двухступенчатой процедуры обеспечивает сохранение резкости границ. Однородные области, выделенные на основе локальных значений стандартных отклонений, обрабатываются далее для обеспечения наилучшего визуального восприятия на последовательных этапах предварительной и усредненной фильтрации. Эксперимент подтвердил получение хороших результатов при общем упрощении процедуры обработки изображений, что важно с точки зрения возможностей аппаратного воплощения.

Ключевые слова: *снижение шумов изображения, выделение границ, сохранение резкости, гауссова фильтрация.*

Directional smoothing model-based image denoising algorithm

© 2020 Xiaoming Zhao, PhD(Physics); Yashuo Bai, postgraduate student; Xin Liu, postgraduate student; Miao Gao, engineer; Kun Cheng, engineer; Shengcun Ma, engineer; Lei Dong, engineer

Xidian University, School of Physics and Optoelectronic Engineering, Xi'an city, China

E-mail: baiyashuo@126.com

Submitted 02.01.2020

DOI:10.17586/1023-5086-2020-87-05-63-76

This study focuses on edge preservation and noise smoothing in the process of denoising. To achieve the two aims, the image has to be processed in such a way that the noise is removed to give people a pleasing vision without reducing the perceptibility of edges and details. In the proposed algorithm, edge orientations are taken into account by using directional templates during edge extraction. The results of convolution would be adopted to control coefficients of the corresponding denoising filter. These two steps play a leading role to guarantee the preservation of edges. Finally, flat regions distinguished from edges and details by local standard deviation would be further operated by incorporating the preliminary filtering result and mean

filtering result to make better vision perception. Its great performance with lower complexity is validated by the experiential results, which provides an important opportunity for hardware implementation.

Keywords: *image denoising, edge-detecting, edge-preserving, Gaussian Filtering.*

OCIS code: *100.2000*

1. INTRODUCTION

Digital images are often contaminated with noise during their acquisition and transmission. Therefore, image denoising, the removal of noise from an image is indispensable as a pretreatment for the subsequent image processing.

The goal of image denoising method is to recover the original image from a noisy measurement

$$f(i) = f_S(i) + \eta(i), \quad \textbf{(1)}$$

where $f(i)$ is the observed value, $f_s(i)$ is the "true" value, $\eta(i)$ is the noise perturbation at a pixel i. The simplest way to simulate additive noise effects is to add a Gaussian white noise with zero mean and standard deviations as $\eta(i)$.

According to actual image characteristics and statistical characteristics of noise, a variety of denoising methods have been developed to recover f_s. Principle of image denoising is to recover real image from noised image and obtain useful information from the image. Therefore, the noise in noisy images must be reduced or removed to obtain the original real image. At present, the commonly used methods are mainly divided into three categories. The first approach is based on spatial domain processing. The second one is based on transform domain processing. The last one is convolution neural network denoising methods, which have emerged in recent years.

In spatial domain, a basic idea of denoising is to calculate weighted average values of neighboring pixels, in which Gaussian filtering [1] and bilateral filtering [2] are widely used. Gaussian filtering denoising method is mainly adopted by adjusting the filter parameter values to form local filter templates. Once the parameter values are selected, filter template is fixed. Meanwhile, inadequate denoising effect and fuzzy edges are emerged consequently when filter templates are formed without considering the image information. Nevertheless, bilateral filtering denoising method has a better denoising effect and can retain the edge of images while suppressing noise, since it can take in to consideration the difference in intensity domain as well as the Euclidean distance in spatial domain. These two mutual constraints control the weights that neighboring pixels contribute to the center of filtering. However, unwanted gradient reversal artifacts may appear near strong edges [3, 4]. The non-local means (NL-means) also calculate weighted averages, which not only compare the grey level in a single point but the geometrical configuration in a whole neighborhood [5]. The following influence is that the computational complexity is too high. In addition, bilateral filtering and NL-means filtering involve manually chosen parameters, which provides possibilities to enhance denoising performance [6].

The method is based on transform domain processing needs to transform the image from spatial domains to other domains first. Then the inherent properties of transform domains will be used to process the transform coefficient and reduce the noise. The final step is to carry out inverse transformation to reconstruct the image. Wavelet transform denoising method [7] is the most commonly used and effective method on transform domain. The main idea of wavelet transform denoising method is to decompose the noisy image at the very beginning and get wavelet transform coefficients. Then wavelet transform coefficients are processed by thresholds, such as soft threshold [8], SureShrink threshold based on the principle of minimizing the Stein unbiased estimate of risk [9] and BayesShrink threshold under the assumption that wavelet coefficients obey the wide-text Gaussian distribution [10], to eliminate the noise and retain details in the image as many as possible. Finally, the denoised image is reconstructed by inverse wavelet transform. Other relevant algorithms include Gaussian mixture model, hidden Markov model, ayes least Square-Gaussian scale mixtures [11–13] and block matching 3D (BM3D) [14]. Among them, BM3D represents one of the

best denoising algorithms since it combines NL-means and wavelet threshold denoising. In summary, this kind of denoising method mainly focuses on the complex processing of wavelet transform coefficients. Among these methods, the threshold function is difficult to obtain and may trigger overkill.

In recent years, a new kind of denoising method is proposed based on convolutional neural network. Viren Jain et al. firstly proposed to use convolutional neural network to solve the denoising problem of natural images and to obtain results that are similar to or even better than conventional methods, such as wavelet transform or Markov random field model [15]. Burger et al. proposed to use the multi-layer perceptron (MLP) for image denoising [16]. One shortcoming of this MLP model is that it cannot adapt to the images with different noise levels. If the noisy images with different levels are used for training, the results of specific noise training will not be achieved. Furthermore, deep convolution neural network becomes widely used in image denoising [17–19], because it has the superior abilities of self-learning through the amount of data and it only need to be guided to achieve the desired purpose instead of strictly selecting the characteristics [20]. However, the huge amount of computation and selection of training set are still problems for these algorithms.

Based on the shortcomings of above algorithms in processing effect and complexity, a new method of image denoising is presented. This method combines the convolution and Gaussian smoothing to achieve three goals:

(1) Removing noise emergence in noisy images to create pleasing output images for the human observers.

(2) Maintaining the sharpness of original image edges.

(3) Reducing the computation complexity to ensure a quicker processing speed.

2. PRINCIPLE OF THE PROPOSED ALGORITHM

In this section, the framework of the proposed algorithm is presented. Then, the core of this paper, the directional smoothing model, is set forth at great length. Finally, smoothing correction in flat region is illustrated for further smooth.

2.1. Framework of proposed algorithm

The entire framework of the proposed algorithm is shown in Fig. 1. This study suggests a convolution template of denoising algorithm to protect the edge and smooth the image selectively. The framework can be divided into two important parts. The first one is the design of a directional smoothing model, which depends on directional edge detection and corresponding filter templates based on Gaussian filtering, to confirm which specific direction each pixel is more inclined to and how many the weights of filter results can be. The preliminary denoising

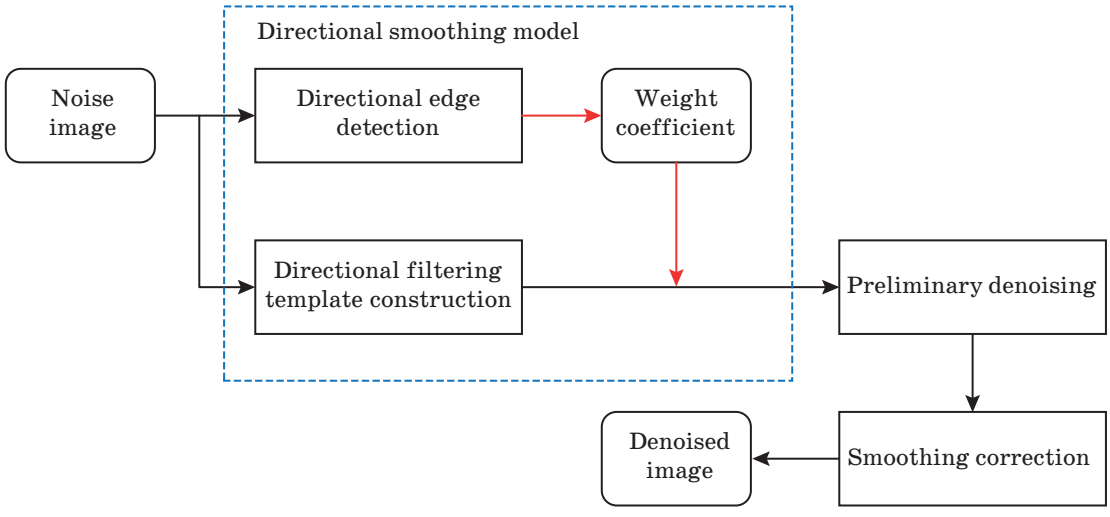

Fig. 1. Framework of proposed algorithm.

is obtained by the process of directional smoothing model as follows:

$$I_{\text{pre}}(i,j)=\sum_{k=1}^{\max(k)}\omega_k(i,j)\,I_k(i,j), \qquad (2)$$

where $I_{\text{pre}}(i, j)$ is the primary denoising result of pixel (i, j), $\omega_k(i, j)$ denotes the weight of k^{th} direction and $I_k(i, j)$ represents the filtering result of k direction.

The second part is smoothing correction, which is aimed at more ideal smoothing results in flat region. Once a pixel is identified in flat area, mean filtering result will be taken into account to get a further smoothing result. The result of denoising is managed as follows:

$$I_D(i,j)=$$
$$=\begin{cases}(1-t)\,I_{\text{pre}}(i,j)+t\,I_{\text{mean}}(i,j), & (i,j)\in \text{flat area}\\ I_{\text{pre}}(i,j), & (i,j)\notin \text{flat area}\end{cases}, \qquad (3)$$

where I_D denotes the output of denoised image, and I_{mean} is the result of mean filtering of pixel (i, j), t controls the weight primary denoising result. Mean filtering result is used to determine gray level of pixel (i, j).

2.2. Design of directional smoothing model

The core of the directional smoothing model is to take the edge direction as the standard to carry out the smoothing on the non-edge direction. The distinguishing of edge pixels and the conduct of preliminary smoothing process greatly depend on two important kinds of templates: directional edge detection templates and smoothing filter templates. They are described in details in the following Section 2.2.1 and Section 2.2.2: directional edge detection and preliminary smoothing.

2.2.1. Directional edge detection

In our designed model, we extract edges directly instead of analyzing differences between edges and flat area in their representation of pixel value or coefficient in other domain and giving a protection to edges. As we all know, edge of a grayscale image is where the grayscale value of a pixel changes, which are usually in the form of a step change or a roof change. The more edge pixels are detected; the more possibility they can be protected. Traditional edge detection methods are based on spatial computation with the spatial differential operators. We propose a mode of edge detection with directional template of variable size based on Prewitt operator.

Various sizes of directional templates are alternatives to meet the different size and scene of images such as 3×3, 5×5, 7×7 and so on. Here we use 5×5 template as an example. If we take full advantage of the 5×5 template, eight directions will be detected and eight convolutions will be designed. When the size of directional templates is 5×5, they can be set as examples in the following form:

$$\text{cov}1=\begin{bmatrix}0&0&0&0&0\\1&1&1&1&1\\0&0&0&0&0\\-1&-1&-1&-1&-1\\0&0&0&0&0\end{bmatrix}$$

$$\text{cov}2=\begin{bmatrix}0&1&0&-1&0\\0&1&0&-1&0\\0&1&0&-1&0\\0&1&0&-1&0\\0&1&0&-1&0\end{bmatrix}$$

$$\text{cov}3=\begin{bmatrix}0&0&0&1&0\\0&0&1&0&-1\\0&1&0&-1&0\\1&0&-1&0&0\\0&-1&0&0&0\end{bmatrix}$$

$$\text{cov}4=\begin{bmatrix}0&1&0&0&0\\-1&0&1&0&0\\0&-1&0&1&0\\0&0&-1&0&1\\0&0&0&-1&0\end{bmatrix}$$

$$\text{cov}5=\begin{bmatrix}0&0&1&0&-1\\0&0&0&0&0\\0&1&0&-1&0\\0&0&0&0&0\\1&0&-1&0&0\end{bmatrix}$$

$$\text{cov}6=\begin{bmatrix}0&0&0&0&1\\0&0&1&0&0\\1&0&0&0&-1\\0&0&-1&0&0\\-1&0&0&0&0\end{bmatrix}$$

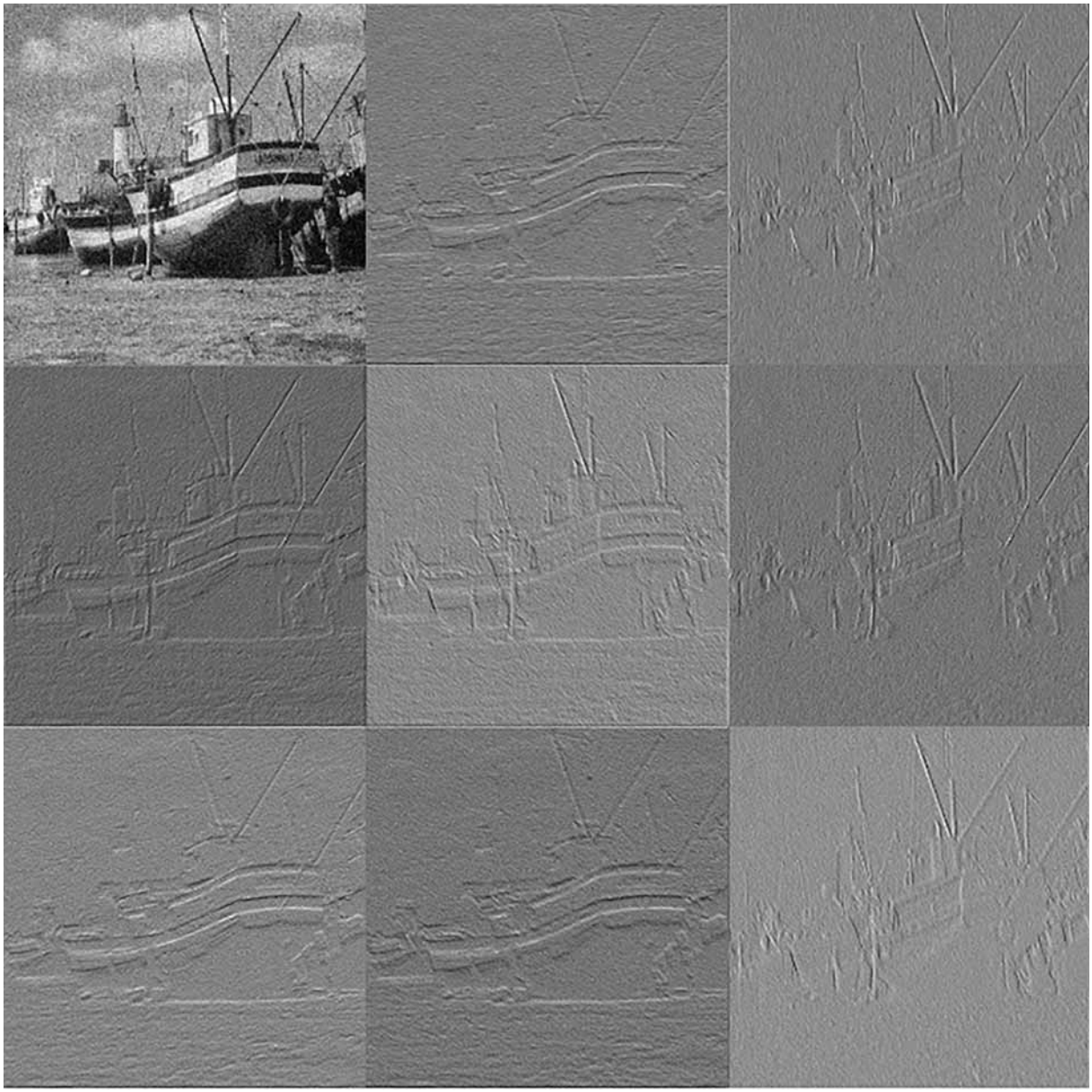

Fig. 2. Example of the convolution results of 5×5 directional templates.

$$\mathrm{cov}7=\begin{bmatrix}1&0&0&0&0\\0&0&1&0&0\\-1&0&0&0&1\\0&0&-1&0&0\\0&0&0&0&-1\end{bmatrix}$$

$$\mathrm{cov}8=\begin{bmatrix}-1&0&1&0&0\\0&0&0&0&0\\0&-1&0&1&0\\0&0&0&0&0\\0&0&-1&0&1\end{bmatrix}. \qquad (4)$$

Obviously, the above convolution templates cov1, cov2, cov3, cov4, cov5, cov6, cov7, and cov8 are used to judge the horizontal edge, the vertical edge, the 45° edge, the 135° edge, the 22.5° edge, the 67.5° edge, the 112.5° edge, and the 157.5° edge separately. The obtained convolution results are defined as E_k, and k is the serial numbers of filter templates which are decided by the size of directional templates. Fugure 2 illustrates the convolution results of 5×5 directional templates.

Cov1 and cov2 use five pairs of pixel point information; cov3 and cov4 use four pairs of pixel point information; cov5, cov6, cov7 and cov8

only use three pairs of pixel point information. Consequently, it is necessary to conduct a normalization processing. For a pixel point (i, j), the following processing is required

$$E'_k(i,j) = E_k(i,j) / s, \tag{5}$$

where s is the amount of pixel point pairs the convolution used and E'_k is the normalized pixel intensity of E_k.

2.2.2. Preliminary smoothing

On the basis of the calculated and normalized convolution results, the filter template that produced by one-dimensional Gaussian formula is weighted accordingly. At every single edge detection direction, one-dimensional Gaussian filtering is put into the filter template under the same size of directional templates. We use this kind of filter template to receive the filtering results. Every single directional edge is protected and the other portions are smoothed by the smoothness property of the Gaussian. If the position of the template is used to be closer to the filtering center, the filtering effect will be smaller.

$$G(r) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-r^2 / 2\sigma^2\right), \tag{6}$$

$$I_k(i,j) = I_{ori}(i,j) g_k. \tag{7}$$

Here, r defines the Euclidean distance between filtering center and weight position. The filter template is denoted as g_k in the formula. Next, filter templates are given as examples with the degrees of 0, 45, and 67.5

$$g_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0.0545 & 0.2442 & 0.4026 & 0.2442 & 0.0545 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$g_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0.0103 \\ 0 & 0 & 0 & 0.2076 & 0 \\ 0 & 0 & 0.5642 & 0 & 0 \\ 0 & 0.2076 & 0 & 0 & 0 \\ 0.0103 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$g_6 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0705 \\ 0 & 0 & 0.8590 & 0 & 0 \\ 0.0705 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{8}$$

In order to use filter templates with different directions more effectively, weights should be assigned to filter templates, so that the background noise can be smoothed on the condition of maintaining the directional edge information. The weight coefficient distribution method of each filter template is as follows: to contrapose one pixel (i, j), if $E'_k(i, j)$ in every direction detected in the previous step is equal to 0, the weight of $I_k(i, j)$ will be divided equally by 1. And the weight is calculated by

$$w'_k(i,j) = E'_k(i,j) \Big/ \sum_{k=1}^{\max(k)} E'_k(i,j). \tag{9}$$

According to the above weight calculation formula, the larger the numerator is, the larger the calculated weight will be. It indicates that the weight distribution mainly depends on the numerator that represents the possible existence of some kind of edge. As the likelihood of the existence increases, the weight of the corresponding edge-direction filtering template will also rise. Therefore, the filtering can suppress the noise while retaining edges. The weight values calculated above are transformed by nonlinear mapping to widen the gap between weight coefficients. The specific nonlinear mapping process is

$$w_k(i,j) = \left(w'_k(i,j)\right)^{\gamma} \Big/ \sum_{k=1}^{\max(k)} \left(w'_k(i,j)\right)^{\gamma}. \tag{10}$$

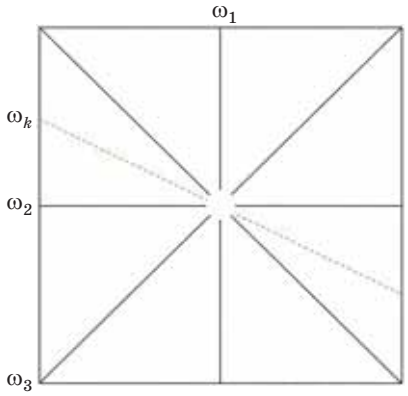

Fig. 3. Directional smoothing sketch map.

Here, γ is adjustable. When $\gamma = 1$, weight coefficients are not stretched. Hence, γ should be given a value that is more than 1 to provide a more effective protection to edges. The directional smoothing sketch map is given in Fig. 3.

Through the directional smoothing model, we detect the pixels on the edges and control the direction of Gaussian smoothing. In accordance with characteristics of one-dimensional Gaussian filtering, edges are preserved well as a result and other regions are received a preliminary smoothing. The processin effect can be seen in Fig. 4.

2.3. Smoothing correction

Through the directional smoothing model which focuses on edge-preserving, a simple denoised effect has been achieved. However, smoothing correction is necessary on account of the fact that smoothing effect is not ideal, because Gaussian templates used in directional smoothing don't offer enough smoothing in nonmarginal region. Local standard deviation (LSD) and threshold T are adopted to make the flat area smoother. The specific correction formula is

$$I_D(i,j) = \begin{cases} (1-t(i,j))\, I_{\text{pre}}(i,j) + t(i,j)\, I_{\text{mean}}(i,j), & \text{LSD}(i,j) \le T \\ I_{\text{pre}}(i,j), & \text{LSD}(i,j) > T \end{cases}, \quad \textbf{(11)}$$

where I_D is the denoised image of the final output; I_{mean} refers to the image obtained after the mean filtering; I_{pre} is the result of directional smoothing model and t is the coefficient obtained according to the LSD value and threshold T is used to distinguish edges and flat areas and determine how many details we retain directly and the bigger t makes smoother result. Here we choose T as 25% of $(\text{LSD})_{\max}$

$$t(i,j) = (T - \text{var}(i,j))^3 / 0.5(T - \text{Min})^3 + 0.5. \quad \textbf{(12)}$$

The role of t is to ensure the smooth transition between smooth region and non-smooth region by controlling weight of result of mean filtering and directional smoothing model.

Figure 5 is a contour plot that demonstrates the value of t. White areas surrounded by blue curves are details and edges. This means our smoothing correction maintains the edges and details after the process of directional smoothing model. Green areas and yellow areas are flatted areas according to our smoothing algorithm. Areas with more yellow pixels represent smoother areas. On the contrary, the more details can be reserved in the areas with more green pixels.

(a)

(b)

Fig. 4. Example of preliminary processing result with directional smoothing model. (a) noised image, (b) preliminary smoothing result.

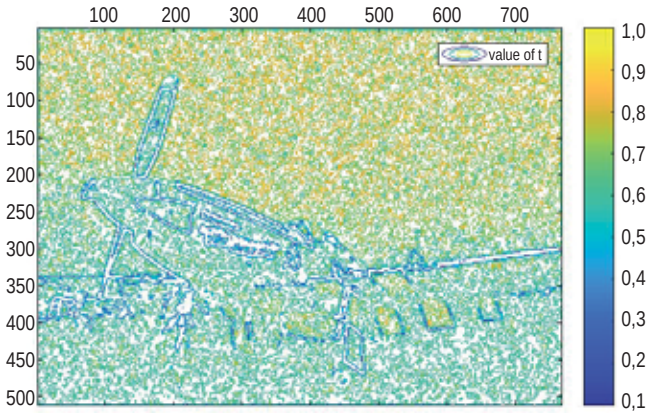

Fig. 5. Contour plot of value t.

3. EXPERIMENTAL RESULTS

In order to validate the proposed approach in denoising, we acquired a comparison in the operation effect of other three denoising technologies, including NL-means [5], bilateral filtering [2], and BM3D [14]. In this section, subjective assessment reflects human visual perception and objective assessment uses peak signal-to-noise ratio (PSNR), structural similarity (SSIM) index [21] and gradient magnitude similarity deviation (GMSD) [22] to give a quantitative analysis on the performance of the four algorithms.

3.1 Subjective assessment

In this section, we use noised images with standard deviation 25 to test general denoising effect among four algorithms. In the last group, the standard deviation is set to be 50. A comparison of denoising effect is revealed to test performance of four algorithms when noise is more serious.

The first sample image shown in Fig. 6 is chosen as the representative of images with obvious directions details. These four algorithms show similar denoised results in general. Nevertheless, if we enlarge the image, we can find that considering holding directional details, the proposed algorithm gains the effect that is similar with NL-means and BM3D, compared with bilateral filtering. Moreover, we see a better performance on maintaining the contour of tire, and more details are preserved to make the image look like the original one in the top right corner, the area of grass and soil.

In Fig. 7, we can pay our attention to the flat area where targets are concentrated. In other words, we need to take a closer look at the flat area with full of details around. To see the smoothing results conveniently, three versions of enlarged image of inflatable dinghy are provided. We can see that there is defective effect in the result of bilateral filtering. However, the flat area has similar smooth effect and keeps the original gradient effect of grayscale in Fig. 7c, e, f after the processing of NL-means, BM3D and proposed method. Besides, through observation of the original image, we believe that the proposed method has ability to reflect the details of the image.

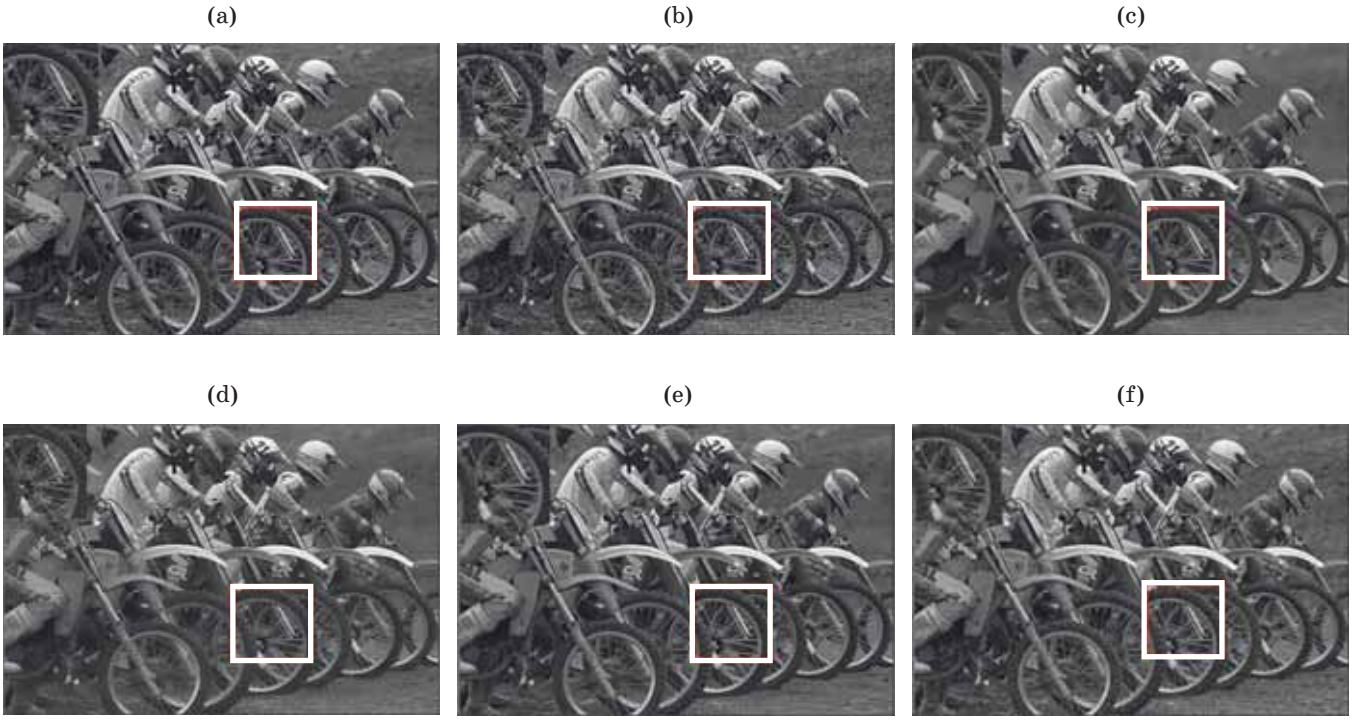

Fig. 6. Output results of three algorithms on “bicycle” image: (a) original image, (b) noised image, (c) NL-means(PSNR = 25.9932, SSIM = 0.9640, GMSD = 0.1121), (d) bilateral filtering(PSNR = 22.6922, SSIM = 0.9237, GMSD = 0.1046), (e) BM3D (PSNR = 27.1347, SSIM = 0.9720, GMSD = 0.0899), and (f) proposed algorithm (PSNR = 24.8803, SSIM = 0.9544, GMSD = 0.0918).

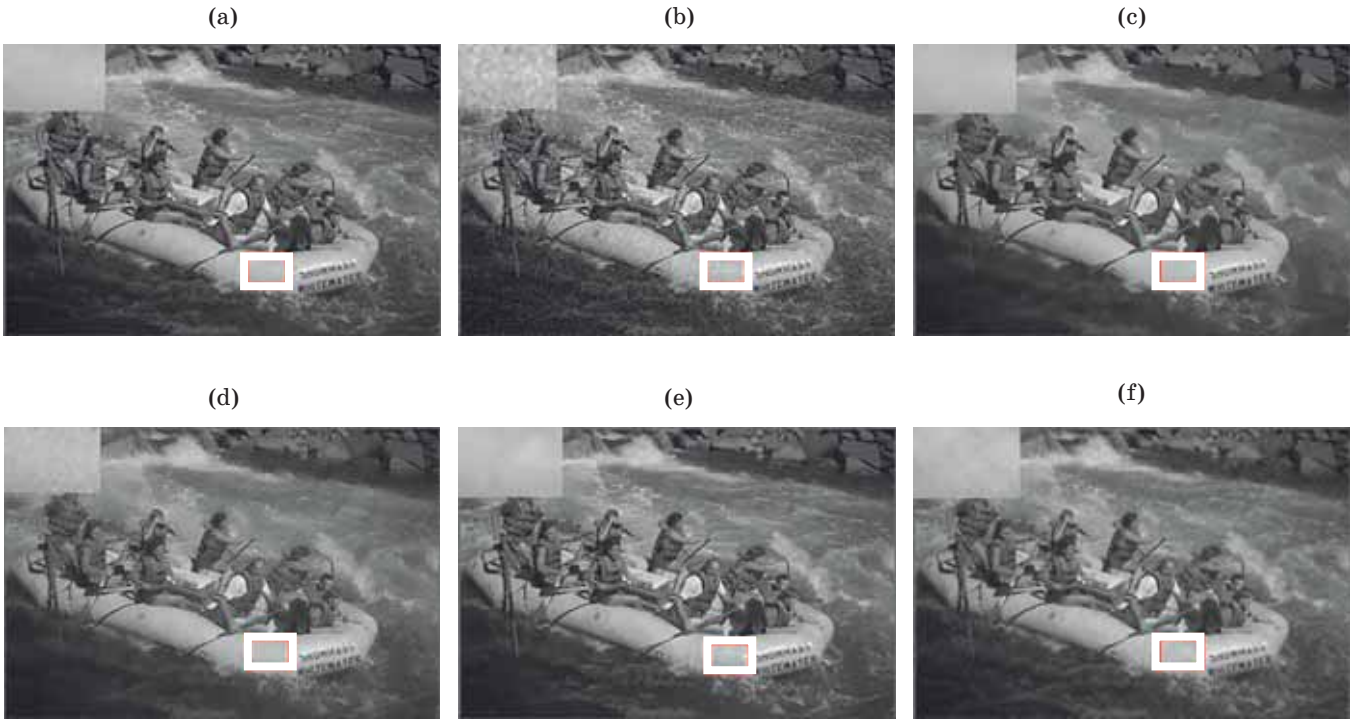

Fig. 7. Output results of three algorithms on "boat" image: (a) original image, (b) noised image, (c) NL-means (PSNR = 27.0126, SSIM = 0.9716, GMSD = 0.1152), (d) bilateral filtering (PSNR = 25.9932, SSIM = 0.9640, GMSD = 0.1121), (e) BM3D (PSNR = 27.8112, SSIM = 0.9771,GMSD = 0.0931), and (f) proposed algorithm (PSNR = 25.8280, SSIM = 0.09616, GMSD = 0.0982).

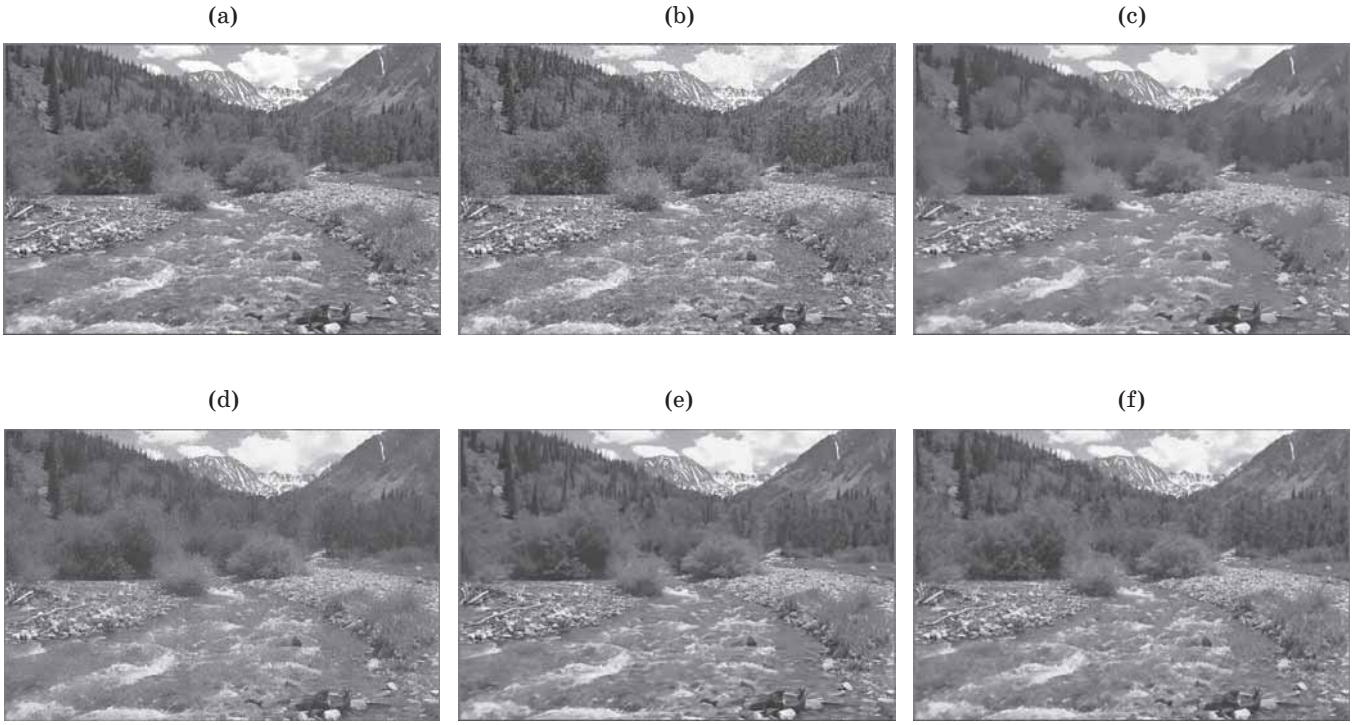

Fig. 8. Output results of three algorithms on "mountain" image: (a) original image, (b) noised image, (c) NL-means (PSNR = 24.2164, SSIM = 0.0.9527, GMSD = 0.1252), (d) bilateral filtering (PSNR = 23.0585, SSIM = 0.9406, GMSD = 0.1076), (e) BM3D (PSNR = 24.8665, SSIM = 0.9597, GMSD = 0.1007), and (f) proposed algorithm (PSNR = 23.9916, SSIM = 0.9512, GMSD = 0.0964).

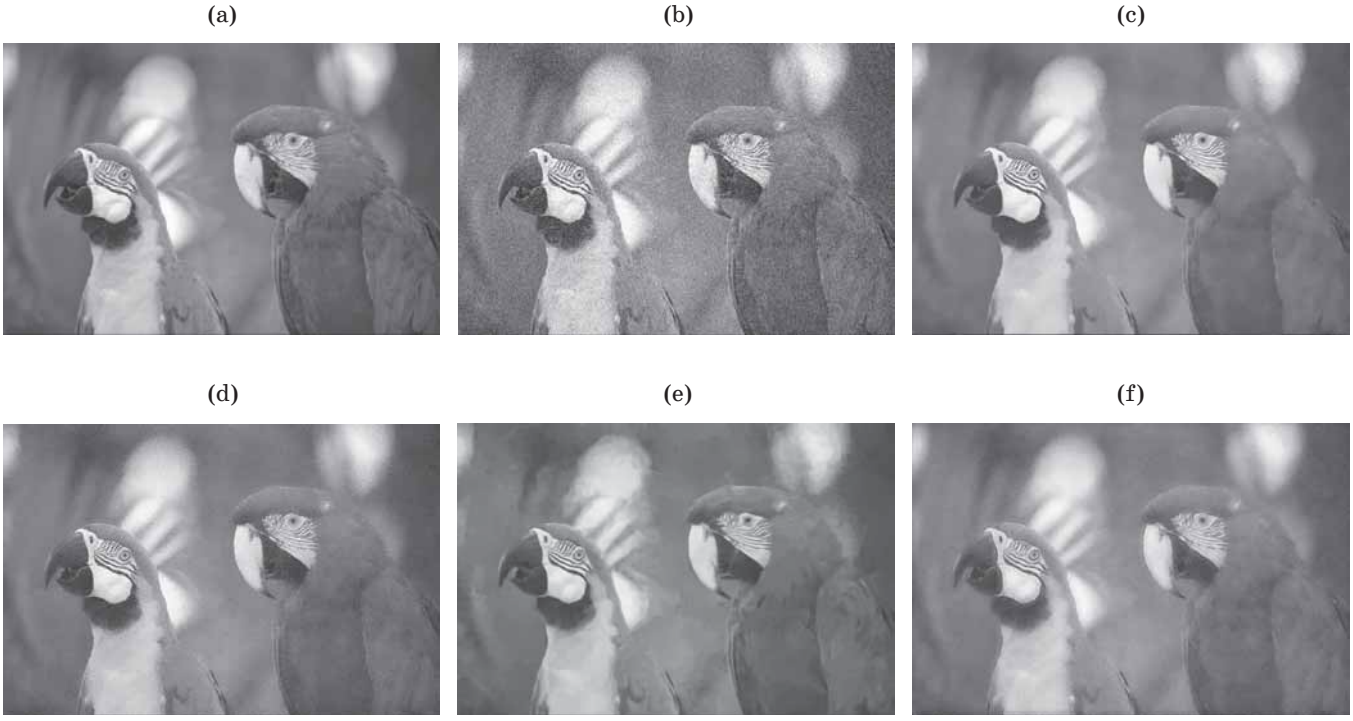

Fig. 9. Output results of three algorithms on "parrot" image: (a) original image, (b) noised image, (c) NL-means (PSNR = 25.5366, SSIM = 0.9849, GMSD = 0.0760), (d) bilateral filtering (PSNR = 26.2454, SSIM = 0.9657, GMSD = 0.0887), (e) BM3D (PSNR = 29.8495, SSIM = 0.9837, GMSD = 0.1007), and (f) proposed algorithm (PSNR = 27.0612, SSIM = 0.9818, GMSD = 0.0792).

Figure 8 is provided to illustrate a general denoising effect of four algorithms. Overall, the more favorable smooth result of image is produced by BM3D and the proposed algorithm to give people a more pleasing vision, compared with bilateral filtering and NL-means. Furthermore, the most details are lost in the area of the river, trees and mountain in the image after it is produced by NL-means, even though the denoising effect is fairly good. As a result, the proposed algorithm is an effective denoising method.

In Fig. 9, denoising performance of the four algorithms is compared under the stronger noise caused by Gaussian noise with a standard deviation of 50. It seems that NL-means perform best in Fig. 9c, keeping really defined edges and clean background with less lose of details. However, the result of BM3D has the problem that edges become unsharp and obvious chunking effect can be noticed. However, Bilateral filtering result still shows grainy effect. In the end, our proposed method in Fig. 9f, on the basis of denoising, can still maintain details well.

3.3. Objective Assessment

Image processing and image reconstruction always focus on making better images. However, it is controversial to define "better" relying entirely on subjective assessment. Objective approach is necessary to determine quantitatively how well the task is performed. Experimental subjects here are all 24 images from the Kodak Lossless True Color Image Suite for objective assessment [23].

As a statistical parameter, PSNR is commonly used in the objective way to assess image fidelity. It is usually used to evaluate the quality of an image after compression of the original image. The larger values indicate less distortion. Although sometimes it is not consistent with the subjective way, it can still reflect the degree of image distortion to some extent. This evaluation index can be calculated by the following formula:

$$\mathrm{MSE} = \frac{1}{HW}\sum_{i=1}^{H}\sum_{j=1}^{W}\left(X(i,j) - Y(i,j)\right)^2, \qquad (13)$$

$$\mathrm{PSNR} = 10\log_{10}\left(\frac{(2^n-1)^2}{\mathrm{MSE}}\right). \qquad (14)$$

Here, MSE represents the mean square error of the current image X and the reference image Y, H and W are the height and width of the image respectively, n is the bits number per pixel which usually values 8.

In addition, SSIM [21] is adopted to measure the structural similarity of two images on bright, contrast and structure. Because of this, structural similarity is more consistent with the human eye's judgment of image quality in the measurement of image quality. The SSIM of two images is calculated by following formulas:

$$l(X,Y)=\frac{2\mu_X\mu_Y+C_1}{\mu_X{}^2+\mu_Y{}^2+C_1}, \quad (15)$$

$$c(X,Y)=\frac{2\sigma_X\sigma_Y+C_2}{\sigma_X{}^2+\sigma_Y{}^2+C_2}, \quad (16)$$

$$s(X,Y)=\frac{\sigma_{XY}+C_3}{\sigma_X\sigma_Y+C_3}, \quad (17)$$

where μ_X, μ_Y means the mean of images X and Y respectively and σ_X, σ_Y are the standard deviations. Meanwhile, σ_{XY} is the covariance. C_1, C_2, C_3 are constants to avoid having a zero in the denominator under a normal value that $C_1 = (K_1{\times}L)^2$, $C_2 = (K_2{\times}L)^2$, $C_3 = C_2/2$ and $K_1 = 0.01$, $K_2 = 0.03$, $L = 255$. Finally

$$\mathrm{SSIM}(X,Y)=l(X,Y)c(X,Y)s(X,Y). \quad (18)$$

The SSIM value range is [0, 1]. The larger the value is, the smaller the image distortion will be.

In Figs. 10 and 11, the average PSNR and SSIM values are plotted, according to different noise standard deviations of 24 images from the Kodak Lossless True Color Image Suite. An overall decline is shown from three compared algorithms as the noise standard deviation increases. However, the proposed algorithm has a better performance than bilateral filtering and is able to approach processing effect of NL-means. Compared with the other three algorithms, BM3D has good capacity under low noise standard deviation. Nevertheless, with increase of noise standard deviation, all algorithms, especially SSIM, have a sharp decline and get a fairly poor value. In general, it is indicative that the proposed algorithm has made a proper denoised effect.

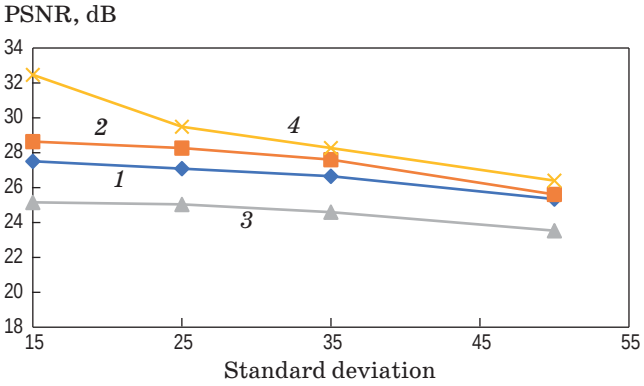

Fig. 10. Comparison of average PSNR of three algorithms. *1* — proposed algorithm, *2* — NL-means, *3* — bilateral filter, *4* — BM3D.

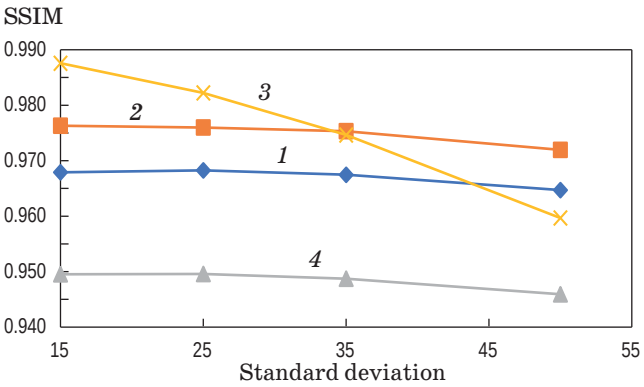

Fig. 11. Comparison of average SSIM of three algorithms. *1* — proposed algorithm, *2* — NL-means, *3* — bilateral filter, *4* — BM3D.

Considering the edge damage caused by denoising operation and the high sensitivity of gradient on image distortion, evaluation on the edge retention of the algorithm is indispensable. We use GMSD [22] to assess the quality of the denoised image. The GMSD of two images is calculated by following formulas:

$$h_x=\begin{bmatrix}1/3 & 0 & -1/3\\ 1/3 & 0 & -1/3\\ 1/3 & 0 & -1/3\end{bmatrix},$$

$$h_y=\begin{bmatrix}1/3 & 1/3 & 1/3\\ 0 & 0 & 0\\ -1/3 & -1/3 & -1/3\end{bmatrix}, \quad (19)$$

$$\mathrm{GMS}(i)=\frac{2m_r(i)m_d(i)+c}{m_r^2(i)+m_d^2(i)+c}. \quad (20)$$

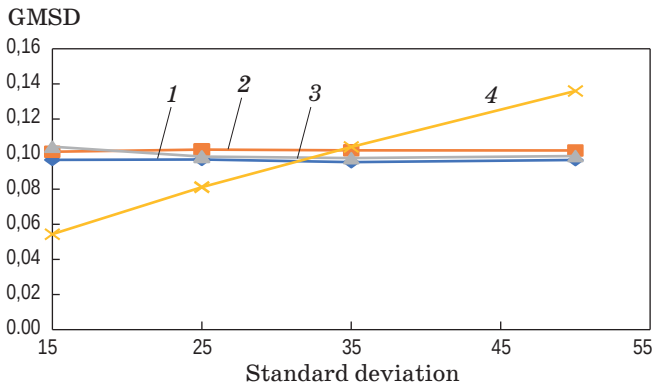

Fig. 12. Comparison of average GMSD of three algorithms. *1* — proposed algorithm, *2* — NL-means, *3* — bilateral filter, *4* — BM3D.

Here, h_x, h_y are used to convolute the image and get the gradient magnitude of horizontal and vertical directions and the results are defined as r and d, i is the location of pixel. After the calculation of the formulas, the local gradient field of each small patch can be calculated. If such a local gradient field is directly used for average processing, it can also reflect the change of image quality, which is called GMSM. And in this method, the deviation of GMS in gradient filed is used to make evaluation more suitable. A lower GMSD means higher quality.

$$\mathrm{GMSD} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\mathrm{GMS}(i) - \mathrm{GMSM})^2}. \quad (21)$$

Figure 12 shows the average GMSD values of proposed algorithm, NL-means, bilateral filtering and BM3D. We can see that the average GMSD of proposed algorithm is lower than that of NL-means and bilateral filtering. Gradient magnitude similarity deviation of BM3D also performs a tremendous change with increase of noise standard deviation. In summary, this means our proposed method has a better edge retention ability.

In Table 1, PSNR, SSIM and GMSD results of five images illustrate the variation trend and overall denoising effect of four algorithms. Just like Fig. 10, in Fig. 11 and Fig. 12, a general decline of denoising effect can be seen with the growth of noise standard deviation. Among four algorithms, BM3D has a more outstanding results compared

Table 1. PSNR, SSIM and GMSD comparison of five runs

Input image	Standard deviation	NL-means			Bilateral Filter			BM3D			Proposed Method		
		PSNR	SSIM	GMSD	PSNR	SSIM	GMSD	PSNR	SSIM	GMSD	PSNR	SSIM	GMSD
Girl	15	30.0622	0.9952	0.0902	26.425	0.9888	0.0888	33.5859	0.9979	0.0485	28.765	0.9934	0.0835
	25	29.8381	0.9950	0.0923	26.3045	0.9886	0.0879	30.7687	0.9960	0.0769	28.5799	0.9932	0.0825
	35	29.0979	0.9944	0.0906	25.9567	0.9881	0.0882	29.3464	0.9942	0.0995	28.0078	0.9927	0.0829
	50	26.7365	0.9920	0.0899	24.5488	0.9854	0.0884	27.7663	0.9890	**0.1236**	26.3892	0.9901	**0.0892**
Sculpture	15	29.1008	0.9829	0.0952	25.2529	0.9584	0.096	33.4905	0.9938	0.0473	28.3837	0.9780	0.0831
	25	28.8325	0.9827	0.0950	25.1244	0.9583	0.0957	30.098	0.9876	0.0743	28.5799	0.9932	0.0825
	35	27.9366	0.9813	0.0950	24.7238	0.9567	0.0956	28.8307	0.9815	0.0956	28.0078	0.9927	0.0829
	50	25.6471	0.9763	0.0952	23.434	0.9514	0.0960	26.4428	0.9679	**0.1231**	25.5793	0.9719	**0.0918**
Window	15	29.5092	0.9733	0.0865	24.7357	0.92	0.1026	34.3962	0.9915	0.0371	28.6555	0.9574	0.0957
	25	29.1576	0.973	0.0855	24.6229	0.9196	0.1023	31.0203	0.9833	0.0647	27.8618	0.9601	0.096
	35	27.9366	0.9813	0.095	24.7238	0.9567	0.0956	28.8307	0.9815	0.0956	27.1566	0.9604	0.0944
	50	25.6471	0.9763	0.0952	23.434	0.9514	0.096	27.8567	0.0959	**0.1157**	25.8084	0.9581	**0.0920**
Hat	15	30.7239	0.9826	0.0828	26.1942	0.9508	0.0941	35.0489	0.9935	0.0424	29.9006	0.978	0.087
	25	30.3039	0.9826	0.0812	26.071	0.9507	0.0950	31.8595	0.9881	0.0697	29.8814	0.9783	0.0897
	35	29.226	0.9815	0.091	25.6119	0.9498	0.0952	31.1767	0.9839	0.0878	29.1285	0.9742	0.0919
	50	26.4134	0.9785	0.0908	24.0848	0.9467	0.0946	29.6189	0.9763	**0.1129**	29.0679	0.9756	**0.0922**
Average		28.51	0.9830	0.0910	25.0780	0.9576	0.0945	30.6335	0.985313	0.0822	28.1096	0.9780	0.0885

with other algorithms. Proposed method shows a similar performance to NL-means. One thing should be noticed is that the proposed method has prominent and steady performance in GMSD. However, BM3D gets terrible results after noise standard deviation at a high level, which means its ability to retain becomes relatively poor.

4. COMPUTATIONAL COMPLEXITY

In terms of the amount of calculation, the algorithm has its advantages. The convolution template is fixed, and there is no need to perform local transformation according to the image information. In addition, the Gaussian filter template only needs to be adjusted according to the filter parameters, and the calculation amount can mainly be reflected when filtering the weight correction based on the edge information. As a result, proposed algorithm has a smaller computational cost than bilateral filtering, NL-means and BM3D.

Table 2 lists the average execution times of the proposed algorithm, NL-means and Bilateral filtering under the noise standard deviation of 25. The objects of the test are also all the images from Kodak Lossless True Color Image Suite. The sizes of the objects are 768×512. We use a personal computer with a 3.3 GHz central processing unit for this test. It can be seen that our algorithm has obvious advantages in processing speed, compared with other methods, especially NL-means.

Table 2 Execution Time Comparison

Algorithm	Time(s)
Proposed	7.389
NL-means	796.356
Bilateral filter	7.871
BM3D	243.3135

5. CONCLUSION

In this paper, we present an image denoising method mainly based on a directional smoothing model, which is able to distinguish edge pixels and control direction of smoothing process. Unlike other denoised methods, diversified edge detection templates are employed to extract edge information. This operation helps to guide obtaining weights of every directional smoothing result using directional smoothing filter according to one-dimensional Gaussian function. This model guarantees to maintain edges and details. Finally, smoothing correction is applied to gain better denoised results. The proposed algorithm demonstrates good performance. To the best of our knowledge, the PSNR and SSIM results shown in Fig. 10 and 11 to evaluate the ability of denoising are similar to results of NL-means and BM3D, which have more complicated strategies. However, the GMSD results shown in Fig. 12 and execution time comparison results perform well to prove the superiority of edge preserving and processing speed.

REFERENCES

1. *Lindenbaum M., Fischer M., Bruckstein A.* On Gabor's contribution to image enhancement // Pattern Recognition. 1994. V. 27. № 1. P. 1–8.
2. *Tomasi C., Manduchi R.* Bilateral filtering for gray and color images // IEEE Sixth Internat. Conf. Computer Vision. 1998. № 98CH. P. 839–846.
3. *Durand F., Dorsey J.* Fast bilateral filtering for the display of high-dynamic-range images // ACM TOG. 2002. V. 21. № 3. P. 257–266.
4. *Bae S., Paris S., Durand F.* Two-scale tone management for photographic look // ACM TOG. 2006. V. 25. № 3. P. 637–645.
5. *Buades A., Coll B., Morel J.M.* A non-local algorithm for image denoising // IEEE Computer Soc. Conf. Computer Vision. 2005. V. 2. № 7. P. 60–65.
6. *Zhang F., Cai N., Wu J., et al.* Image denoising method based on a deep convolution neural network // IET Image Proc. 2018. V. 12. № 4. P. 485–493.
7. *Coifman R.R., Donoho D.L.* Translation-invariant de-noising // Wavelets & Statistics. 1995. V. 103. № 2. P. 125–150.
8. *Donoho D.L.* De-noising by soft-thresholding // IEEE Trans. Information Theory. 2018. V. 41. № 3. P. 613–627.
9. *Donoho D.L., Johnstone I.M.* Adapting to unknown smoothness via wavelet shrinkage // J. American Statistical Association. 1995. V. 90. № 432. P. 1200–1224.

10. *Chang S.G., Yu B., Vetterli M.* Adaptive wavelet thresholding for image denoising and com-pression // IEEE Trans. Image Proc. 2000. V. 9. № 9. P. 1532–1546.
11. *Chipman H.A., Kolaczyk E.D., McCulloch R.E.* Adaptive Bayesian wavelet shrinkage // J. American Statistical Association. 1997. V. 92. № 440. P. 1413–1421.
12. *Moulin P., Liu J.* Analysis of multiresolution image denoising schemes using generalized Gaussian and complexity priors // IEEE Trans. Information Theory. 1999. V. 45. № 3. P. 909–919.
13. *Romberg J.K., Choi H., Baraniuk R.G.* Bayesian tree-structured image modeling using wavelet-domain hidden Markov models // EEE Trans. Image Proc. 2001. V. 10. № 7. P. 1056–1068.
14. *Dabov K., Katkovnik V., Foi A., et al.* Image denoising by sparse 3D transformation-domain collaborative filtering // IEEE Internat. Conf. Image Proc. 2007. V. 16. № 7. P. 1–16.
15. *Jain V., Seung H.S.* Natural image denoising with convolutional networks // Internat. Conf. Neural Information Proc. Systems. 2008. P. 769–776.
16. *Burger H.C., Schuler C.J., Harmeling S.* Image denoising: Can plain neural networks compete with BM3D? // Computer Vision and Pattern Recognition. 2012. V. 157. № 10. P. 2392–2399.
17. *Wang X., Wang L., Tao Q., et al.* Deep convolutional architecture for natural image denoising // Internat. Conf. Wireless Communications & Signal Proc. 2015. V. 5. № 53. P. 1–4.
18. *Koziarski M., Cyganek B.* Deep neural image denoising // Internat. Conf. Computer Vision and Graphics. 2016. P. 163–173.
19. *Lefkimmiatis S.* Non-local color image denoising with convolutional neural networks // IEEE Conf. Computer Vision and Pattern Recognition. 2017. P. 5882–5891.
20. *Zhang F., Cai N., Wu J., et al.* Image denoising method based on a deep convolution neural network // IET Image Proc. 2018. V. 12. № 4. P. 485–493.
21. *Wang Z., Bovik A.C., Sheikh H.R., and Simoncell E.P.* Image quality assessment: From error visibility to structural similarity // IEEE Trans. 2004. V. 13. № 4. P. 600–612.
22. *Xue W., Zhang L., Mou X., Bovik A.* Gradient magnitude similarity deviation: A highly efficient perceptual image quality index // IEEE Trans. Image Proc. 2014. V. 23. № 2. P. 684–695.
23. *Barrett H.H.* Objective assessment of image quality: Effects of quantum noise and object variability // JOSA. 1990. V. 7. № 7. P. 1266–1278.